A PROJECT REPORT ON

CLASSIFICATION OF GALAXIES USING FASTER RCNN

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY , PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING
(Computer Engineering)

BY

Tejshree Bang	Exam No: 71710166F
Riya Chaddha	Exam No: 71709708M
Kimaya Devasthali	Exam No: 71611746M
Prasad Katore	Exam No: 71611368G

Under the guidance of

Mrs. Archita Patil

DEPARTMENT OF COMPUTER ENGINEERING

PES MODERN COLLEGE OF ENGINEERING
SHIVAJINAGAR,PUNE 411005

SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

2019 - 20

CERTIFICATE

This is to certify that the Project Entitled

CLASSIFICATION OF GALAXIES USING FASTER RCNN

Submitted by

Tejshree Bang	Exam No: 71710166F
Riya Chaddha	Exam No: 71709708M
Kimaya Devasthali	Exam No: 71611746M
Prasad Katore	Exam No: 71611368G

is a bonafide work carried out by them under the supervision of Mrs. Archita Patil and it is approved for the partial fulfillment of the requirement of Savtribai Phule Pune university, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering).

Mrs. Archita Patil
Guide
Department of Computer Engineering

Prof. Dr. Mrs. S. A. ITKAR
Head
Department of Computer Engineering

Signature of Internal Examiner

Signature of External Examiner

PROJECT APPROVAL SHEET

A Project Report Titled as

CLASSIFICATION OF GALAXIES USING FASTER RCNN

is successfully completed by

Tejshree Bang	Exam No: 71710166F
Riya Chaddha	Exam No: 71709708M
Kimaya Devasthali	Exam No: 71611746M
Prasad Katore	Exam No: 71611368G

at

DEPARTMENT OF COMPUTER ENGINEERING

PES MODERN COLLEGE OF ENGINEERING

SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE

ACADEMIC YEAR 2019-2020

Mrs. Archita Patil	Prof.Dr. Mrs. S. A. ITKAR
Guide	Head
Department of Computer Engineering	Department of Computer Engineering

Acknowledgement

It gives us pleasure in presenting the project report on **'CLASSIFICATION OF GALAXIES USING FASTER RCNN'**.

Firstly, we would like to express our indebtedness appreciation to our guide **Mrs. Archita Patil**. Her constant guidance and advice played very important role in successful completion of the project. She always gave us her suggestions, that were crucial in making this report as flawless as possible.

We would like to express our gratitude towards **Prof. Dr. Mrs. S. A. Itkar** Head of Computer Engineering Department, PES Modern College of Engineering for her kind co-operation and encouragement which helped us during the completion of this report.

Also we wish to thank our Principal, **Prof. Dr. Mrs. K. R. Joshi** and all faculty members for their whole hearted co-operation for completion of this report. We also thank our laboratory assistants for their valuable help in laboratory.

Last but not the least, the backbone of our success and confidence lies solely on blessings of dear parents and lovely friends.

Tejshree Bang
Riya Chaddha
Kimaya Devasthali
Prasad Katore

Abstract

Sky survey organizations take hundreds of thousands of images of terrestrial bodies, galaxies and other astronomically important data which is studied by space organizations. To study the astronomical aspect of galaxies, first these galaxies need to be classified. We present a method for automatic detection and classification of galaxies to eliminate the manual work which includes a novel procedure to make a robust trained model. This method is shown for processing and analyzing big astronomical datasets, including high performance Python used in the areas of image processing and computer vision.The model is trained using Faster Region-based Convolutional Neural Network i.e. Faster RCNN which is a deep learning technique. The model was trained using public datasets such as the Sloan Digital Sky Survey (SDSS) and Galaxy Zoo Dataset.

Keywords

Machine Learning, galaxy, object Detection, Galaxy, Galaxy Zoo, SDSS(Sloan Digital Sky Survey), Image Processing, Faster RCNN(Region-based Convolutional Neural Network), deep learning.

Contents

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION

1.1 Overview

Traditionally classification of galaxies was done manually. As Technology improves a lot of astronomical data is being produced. This data is largely unclassified and processed. Our system solves this problem by automatically classifying galaxies by The Hubble Sequence. The system can classify the main types of galaxy that are Elliptical & Spiral according to the Hubble Sequence. It can also classify Edge-on type of galaxies. The dataset is labelled using the concept of Decision Tree where we built a code for the same and all the class labels are stored in a CSV file with their galaxy ids. Data augmentation is used to enhance data and extract features from the image in order to train the model with more efficiency. There are various data augmentation techniques out of which cropping was used to decrease or more effectively eliminate noise from the images. Then an xml file is generated for data which is given as an input to the model as train labels. Along with Faster RCNN algorithm, we are using tensorflow library to effectively train the model. Multiple images can be passed to the model while testing where model predicts the class of galaxy and output is given as an image with bounding boxes and label showing the class to which the galaxy in the image belongs.

1.2 Motivation

1. Sky Surveys like the Sloan Digital Sky Survey take hundreds of thousands of images of galaxies with each passing day. It becomes critical to classify these galaxies. As better and bigger telescopes continue to collect these images, the data-sets begin to explode in size.

2. In order to better understand how the morphology of galaxies relates to the physics that create them, such images need to be sorted and classified. With the help of Machine Learning Methods implemented efficiently this process can execute within seconds.F

3. SDSS will take more than 50 million images of galaxies in the near future as technology improves. Thus it is important to classify them for further research and study.

1.3 Problem Definition and Objectives

To Detect and Classify Galaxies using Faster Region-based Convolutional Neural Networks(Faster RCNN).

1.4 Project Scope & Limitations

1.4.1 Project Scope

1. Detection and Classification of Galaxies using Faster RCNN.
2. Classification of Galaxies i.e. Elliptical, Spiral and Lenticular.
3. There exists a lot of astronomical data. By using this data we can classify all classes and sub-classes of galaxies
4. This will also require the use of system with better performance and graphics processing power.
5. With further training, testing and incorporation of methods and technology developed in the near future the project can be made much more accurate and detailed.

1.4.2 Limitations

1. Data is available in varied formats. It is difficult to bring all the data on the same platform of datatype.
2. Detects the broad type of Galaxies i.e. Elliptical, Spiral & Edge-on. Does not detect more classes of galaxies such as Lenticular, Barred Spiral, Merger etc. This can be achieved with more data.
3. The system accepts images only in .jpg format.
4. Detects main class and not the sub-classes for galaxies i.e. E0, E1, E2 etc for Elliptical, S0, S1, S2 etc for Spiral and so on.
5. Only detects non-overlapping galaxies.

1.5 Methodologies for Problem Solving

We have used top down approach to tackle this problem.A top-down approach is essentially the breaking down of a system to gain insight into its compositional sub-systems in a reverse engineering fashion. In a top-down approach an overview of the system is formulated, specifying, but not detailing, any first-level subsystems. Each subsystem is then refined in yet greater detail, sometimes in many additional subsystem levels, until the entire specification is reduced to base elements.

Applying top down approach involves labelling the data and applying data augmentation. After preprocessing the data, data is split into two sets i.e training and testing dataset. Each module is then combined to generate tf records for training set and this module is given as input to train the model.

CHAPTER 2

LITERATURE SURVEY

Sr. No.	Title	Author	Publication Details	Algorithmic Approach
1	Galaxy detection and identification using deep learning and data augmentation.	Roberto E. Gonźalez, Roberto P. Muñoz, Cristian A. Herńandez.	arXiv.org, September 2018	Training using Deep Learning framework DarkNet and YOLO for object detection.
2	Deep Galaxy V2: Robust Deep Convolutional Neural Networks for Galaxy Morphology Classifications	Nour Eldeen Khalifa, Mohamed Hamed Taha, Aboul Ella Hassanien, Ibrahim Selim	IEEE, June 2018	Deep CNN.
3	Automatic Detection of Galaxy Type From Datasets of Galaxies Image Based on Image Retrieval Approach.	Mohamed Abd El Aziz, I. M. Selim and Shengwu Xiong.	NCBI and Scientific Reports Vol. 25, June 2017	Data Augmentation using binary algorithms and Deep CNN.
4	Neural networks on galaxy pattern recognition.	Ricardo Contreras, Gabriel Salazar, M. Angelica Pinninghoff, Neil Nagar.	8th International Conference of Pattern Recognition Systems 2017	Deep CNN.
5	Galaxy Zoo 2: detailed morphological classifications for 304,122 galaxies from the Sloan Digital Sky Survey	Kyle Willet, Chris Lintott, Steven P. Bamford, Robert C. Nichol, Kevin Schawinski, Robert J. Simpson, Edward M. Edmondson	arXiv:1308.3496-v2 [astro-ph.CO], 19th August, 2013	Decision Tree to identify the class of galaxy.

Table 2.1: Literature Review

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Assumptions and Dependencies

1. Dataset:
   Assumptions: The dataset contains 30k images out of which 'elliptical', 'spiral' and 'edge-on' types of galaxy images must each not be less than 5k.
   Dependencies: The images are already in .jpg format.

2. IDE: Jupyter(Anaconda 2019.03)

3. Python packages: Tensorflow, Sklearn, Pandas, Numpy, Pillow, Pandas, Matplotlib, ntpath, csv, glob.

3.2 Functional Requirements

3.2.1 System Feature 1

System only detects 'Elliptical', 'Spiral' & 'Edge-on' classes. Thus images need to be of these classes.

3.2.2 System Feature 2

The images need to be supported by a CSV file that details the image name and class.

3.2.3 System Feature 3

The output is in the form of image having bound box around the detected galaxy with it's class labelled.

3.3 External Interface Requirements

3.3.1 User Interfaces

3.3.2 Hardware Interfaces

1. CPU : For Data Cleaning and preprocessing.

2. GPU : For Image Processing.

3.3.3 Software Interfaces

1. Operating System : We have chosen Windows for its user friendliness and support for software used for project.
2. Anaconda : To implement our project on Jupyter Notebook as it has an interactive interface to run python scripts.

3.3.4 Communication Interfaces

3.4 Nonfunctional Requirements

3.4.1 Performance Requirements

A highly powerful CUDA compatible GPU will boost the performance and reduce latency extensively, making the system almost real-time.

3.4.2 Safety Requirements

No major safety constraints applicable.

3.4.3 Security Requirements

No major security constraints are applicable to this system as it is Open Source and free to use for all.

3.4.4 Software Quality Attributes

1. Correctness : The model should detect the type of galaxy perfectly.
2. Accuracy : Maximizing the accuracy is the main quality attribute.
3. Speed : Prediction of the galaxy within few milliseconds is a desirable quality attribute.

3.5 System Requirements

3.5.1 Database Requirements

1. In our project, we have used DR(Data Release) 15 which was used by Galaxy Zoo where they prepared and converted dataset which was in fits form to jpg images.

2. The data is stored in CSV which consists of classes of galaxies with their image ids. Seven classes are stored in this CSV file viz. elliptical, spiral, merger, ring, edge-on, barred spiral and irregular.

3. Some classes have proven to have insufficient data for proper classification.Thus Database needs to be segregated and cleaned.

3.5.2 Software Requirements

1. Windows 10
2. Cuda
3. Anaconda 2019.03

3.5.3 Hardware Requirements

1. Processor: Intel® Core
2. System type: 64 bit operating system(x64)
3. Graphics Card: Nvidia GeForce GTX 1050 Ti GPU.

3.6 Analysis Models:SDLC Model to be applied

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

1. Requirements and Analysis:
   The first phase is to understand what we want to develop and what all functions the software should have. Dataset is gathered from Sloan Digital Sky Survey i.e DR15. Software/Hardware requirements are also analysed in this phase.

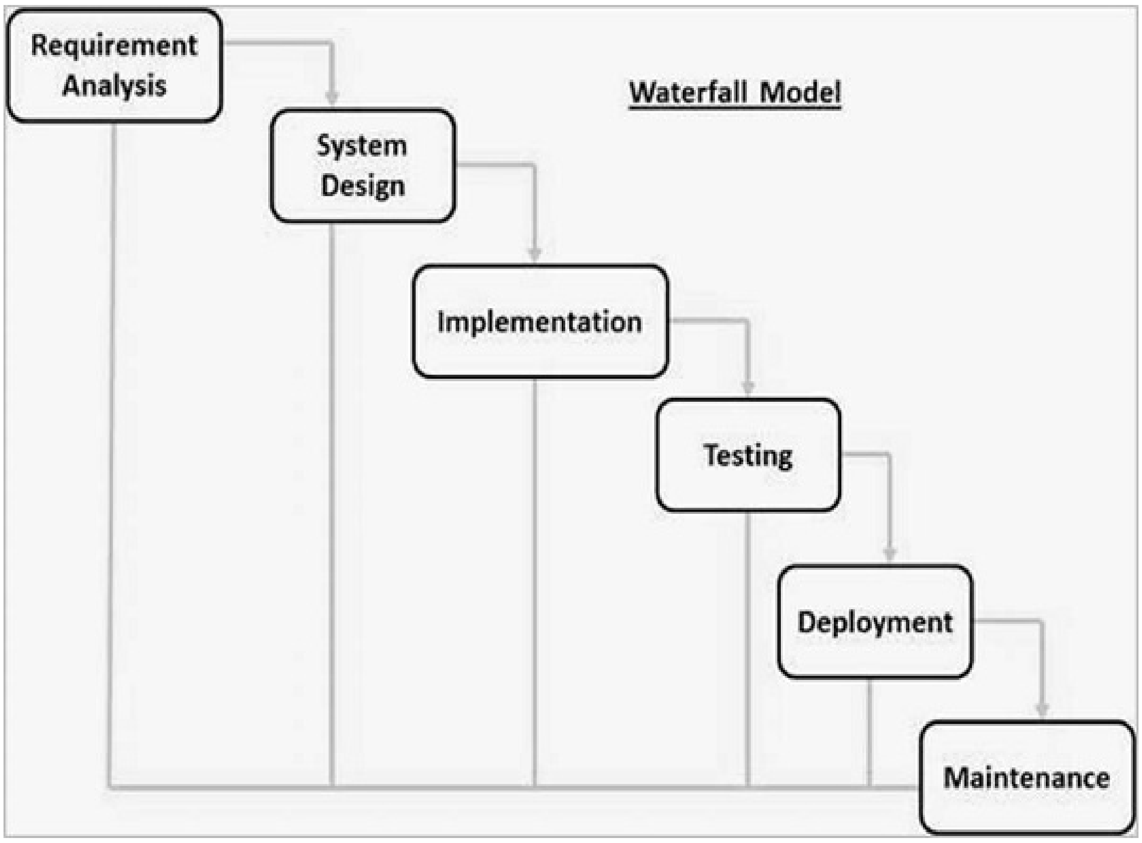


Figure 3.1: Waterfall Model

2. System design :
   In the second phase, architecture of the system is designed. All the different technical questions that may appear in this stage are discussed. Also, the technologies used in the project, team load, limitations, time frames,and budget are discussed. The most appropriate project decisions are made according to the defined requirements.

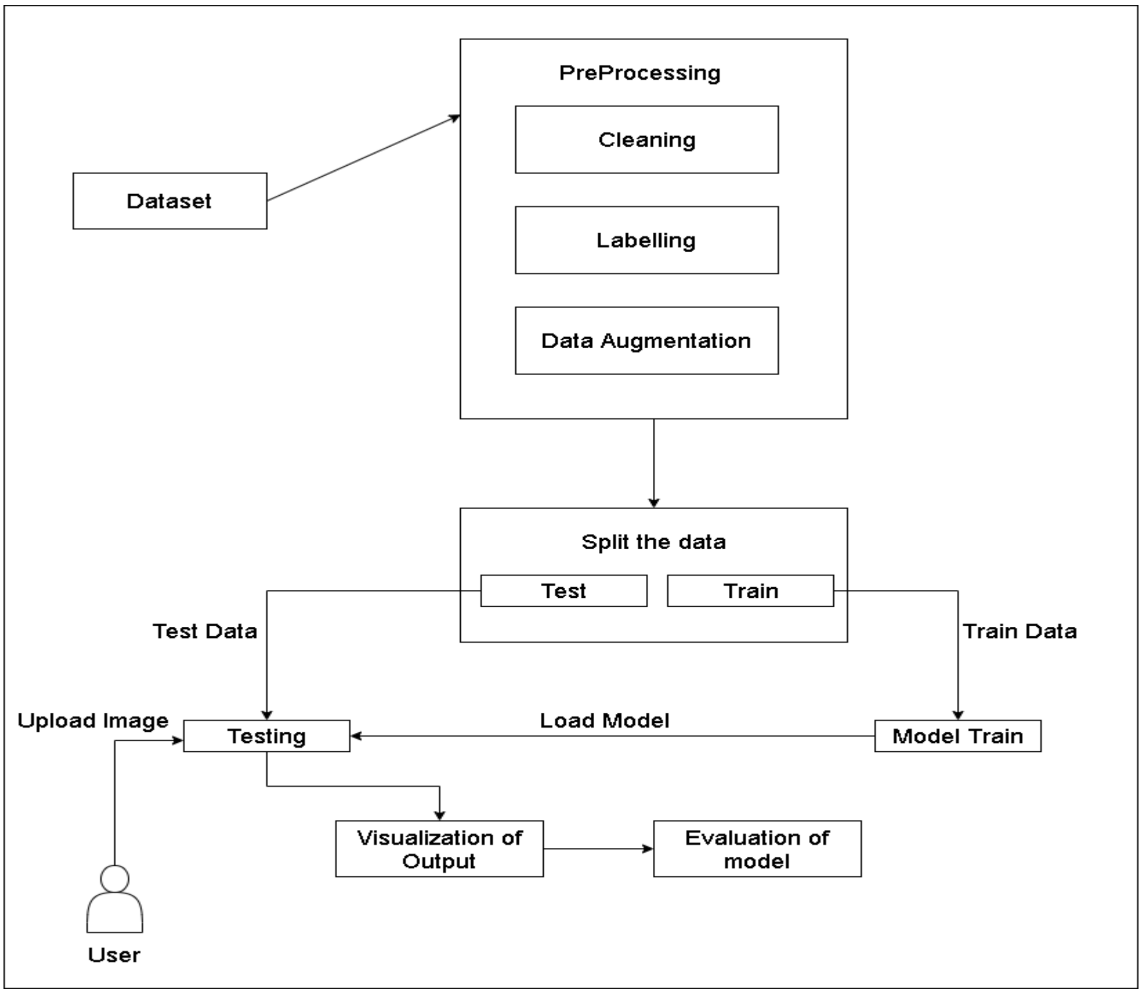


Figure 3.2: System Design

3. Implementation :
   After the requirements are analyzed and system design is approved comes the stage of implementation. Programmers start developing modules of system while keeping requirements in mind that are defined. The system administrators adjust the software environment for

the developers.
The below diagram shows the implementation steps:

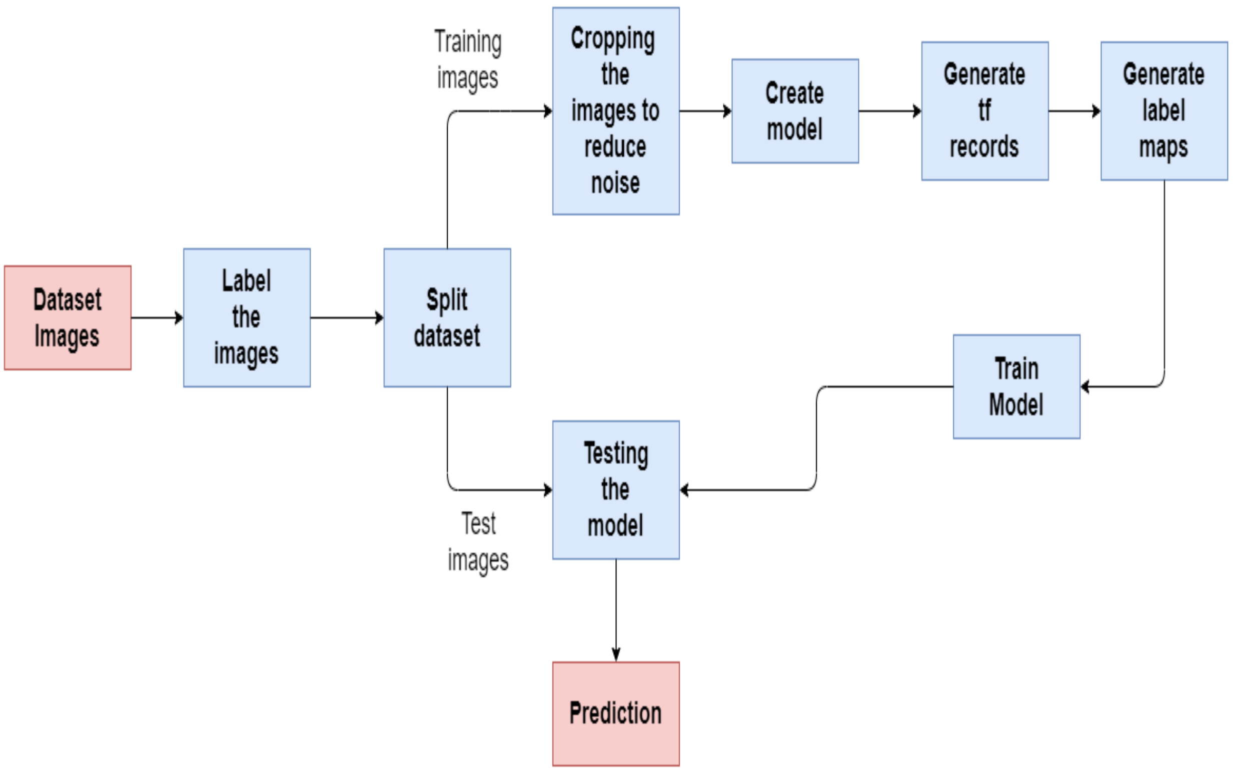


Figure 3.3: Implementation Steps

1. First, images are collected and if they are not in jpg format, are converted in the required format i.e in jpg form.

2. We build a code so that it can label all the images correctly.

3. After labelling, we need to split the dataset into training and testing which will be used for the training and testing process respectively.

4. We pass train images to the cropping module where it will crop out the unnecessary part and we get only the required portion of the image so that only the important features are extracted and we can eliminate noise from the images if any.

5. We create a model in tensorflow where we are loading all the files of faster RCNN algorithm.

6. Generate tf records for the training dataset. It basically loads our data in the model created in form of binary input.

7. Create label maps for classification. In our model, we have generated 3 label maps that are saves elliptical, spiral and edge-on labels.

8. Training the model for upto 5 to 7 hours until the total loss becomes less than 0.05%. If training is done for a long time, after a point model starts to overfit so while training we need to make sure that it does not encounter overfitting.

9. This is the testing part which is done after the model is trained well. Test images are passed in testing file where it loads the model and performs testing and evaluation such as accuracy.

10. This is output which we can see in the form of an image which has bounding boxes around the detected galaxy and also label which predicts which type of galaxy it is.

4. System testing :
   All the units developed in the implementation phase are integrated into a system after testing of each unit.The software designed, needs to go through constant software testing to find out if there are any flaw or errors.

5. System Deployment:
   Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

6. System Maintenance :
   There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

CHAPTER 4
SYSTEM DESIGN

4.1 System Architecture

In the second phase, architecture of the system is designed. All the different technical questions that may appear in this stage are discussed. Also, the technologies used in the project, team load, limitations, time frames,and budget are discussed. The most appropriate project decisions are made according to the defined requirements.
The system architecture is shown below:

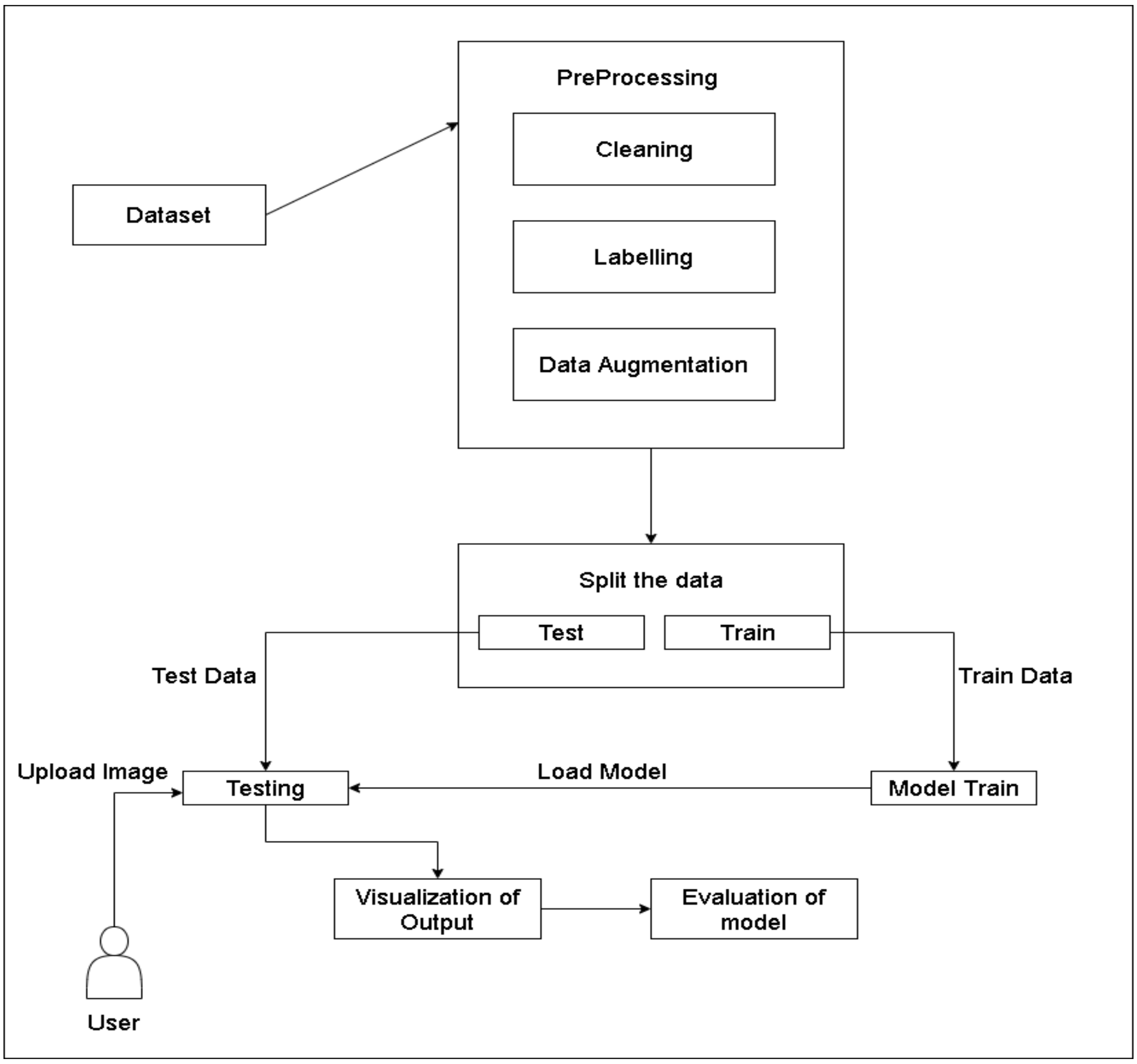


Figure 4.1: System Architecture Diagram

Fig 4.1 illustrates how we prepared our system for prediction of different types of galaxies. The dataset goes through pre-processing where data is cleaned after which the data is labelled using the probabilities of every class. A decision tree is built and run for every image consisting of these class probabilities and the final result i.e. class label is stored in a CSV file. Data augmentation is applied on the data where images are cropped to extract important features of galaxy from the image. There are total 7 class labels out of which this system focuses only on 3 classes which needs to be separated from rest of the images The dataset is then split into training and testing data respectively. Tf records and label maps are generated for training data and classes respectively. The model is trained by using Faster RCNN API and then this model is tested using test dataset to evaluate the accuracy and precision of it.

4.2 Mathematical Model

1. Input : I = {i1, i2, ..., in}
   Input to the system is n number of images.

2. Processing : P = {P1, P2, P3, P4, P5}

   (a) P1 : Convolutional Layers.

      i. Input : I = {i1, i2, ..., in}

      ii. Processing : Feature Map Extraction by passing image through Convolutional layers

      iii. Output : R1 = Feature Maps.

(b) P2 : Region Proposal Network.

   i. Input : R1 = Feature Maps.
   ii. Processing : Sliding Window over each feature for region proposal.
   iii. Output : R2 = Region Proposals.

(c) P3 : ROI Pooling.

   i. Input : R2 = Region Proposals.
   ii. Processing : ROI pooling done based on Region Proposals.
   iii. Output : R3 = ROI Pools.

(d) P4 : Classifier

   i. Input : R3 = ROI Pooling.
   ii. Processing : Classifier uses ROI pooling for Object Classification.
   iii. Output : R4 = Object Classification.

3. Output O : Classification of image into its type of Galaxy.

4.3 Data Flow Diagrams / UML Diagrams

4.3.1 Class Diagram

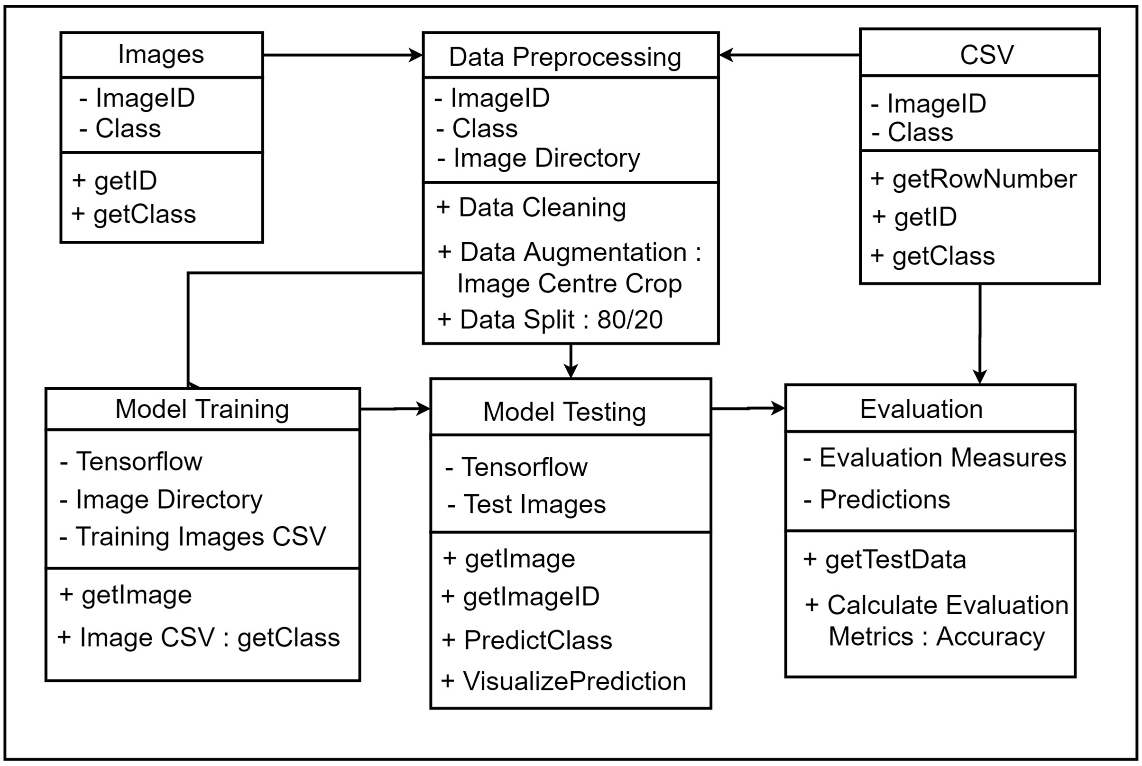


Figure 4.2: Class Diagram

4.3.2 Use Case Diagram

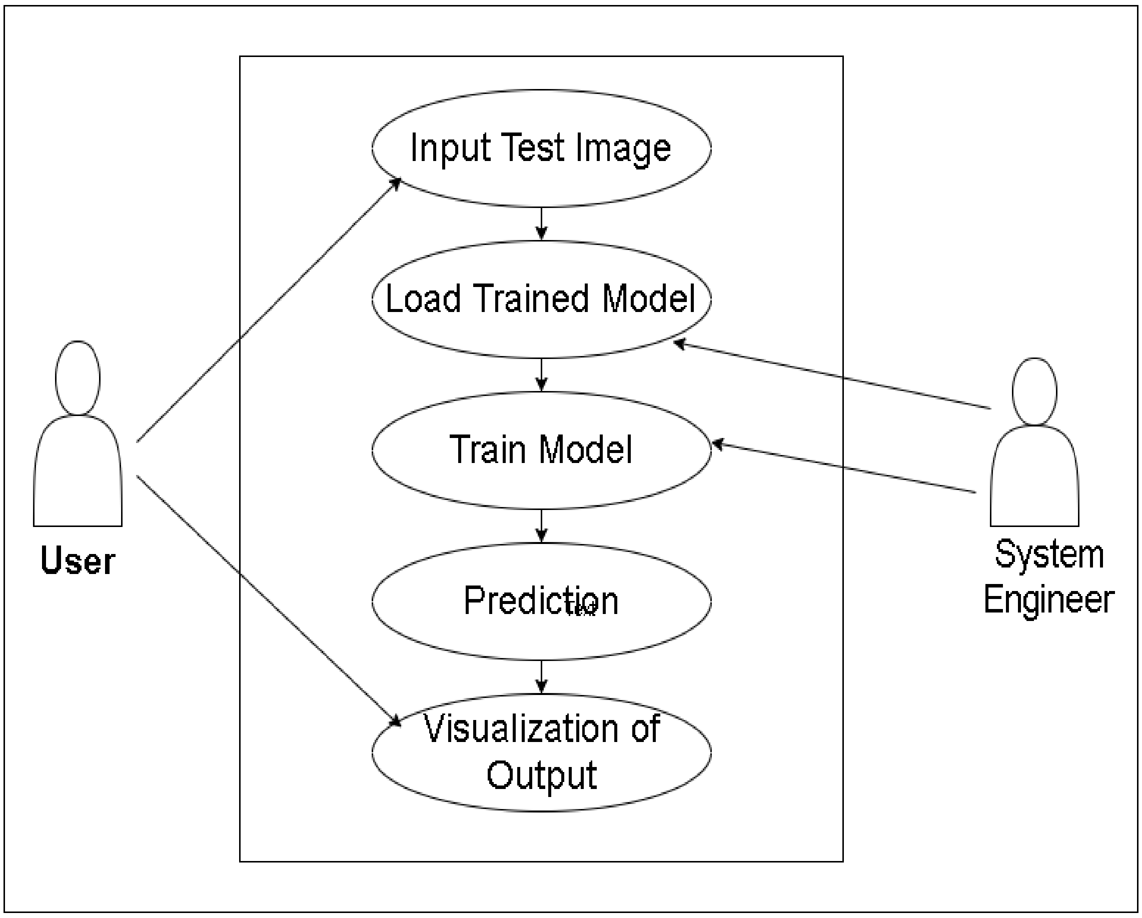


Figure 4.3: Use Case Diagram

4.3.3 Package Diagram

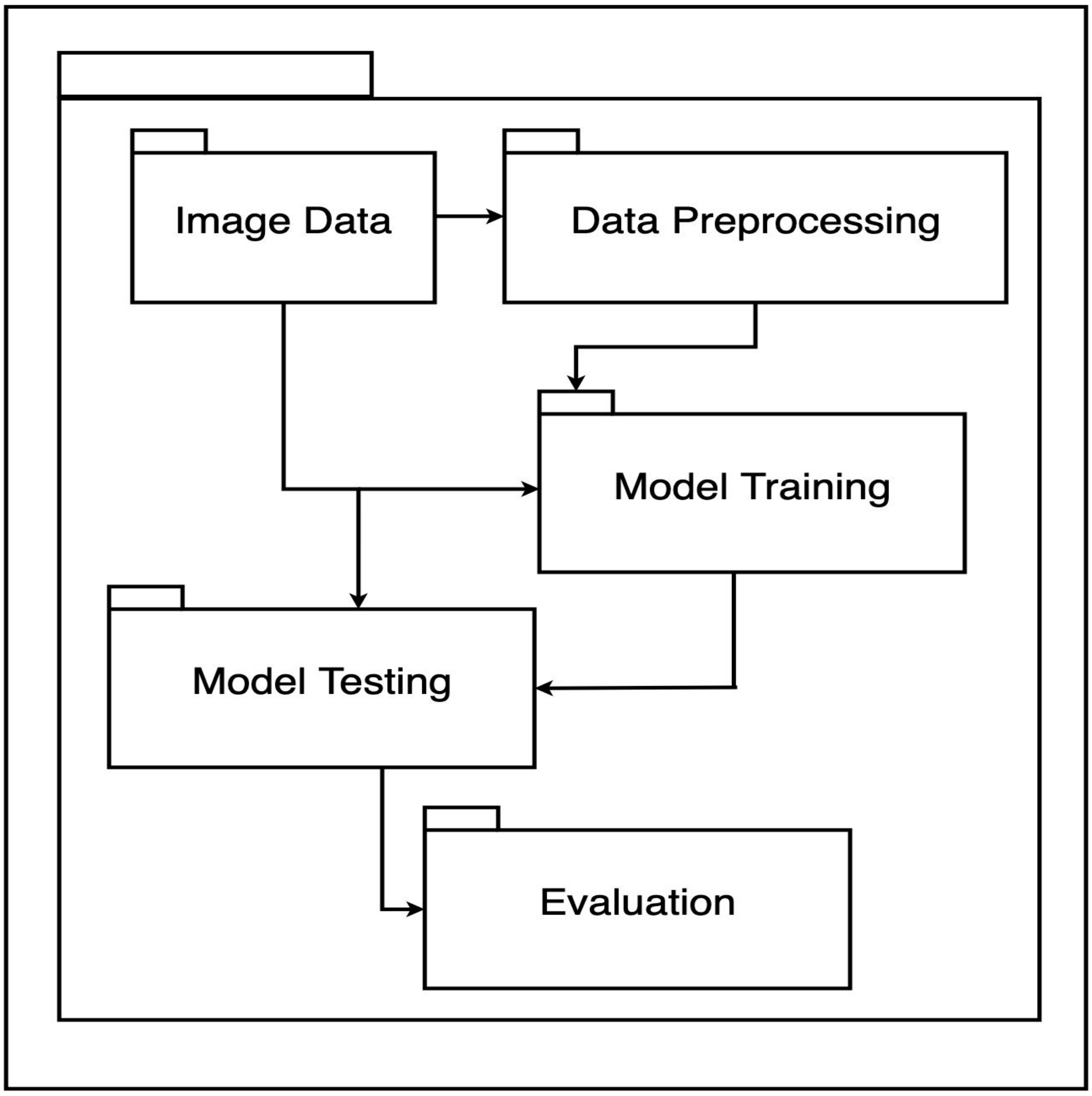


Figure 4.4: Package Diagram

4.3.4 Component Diagram

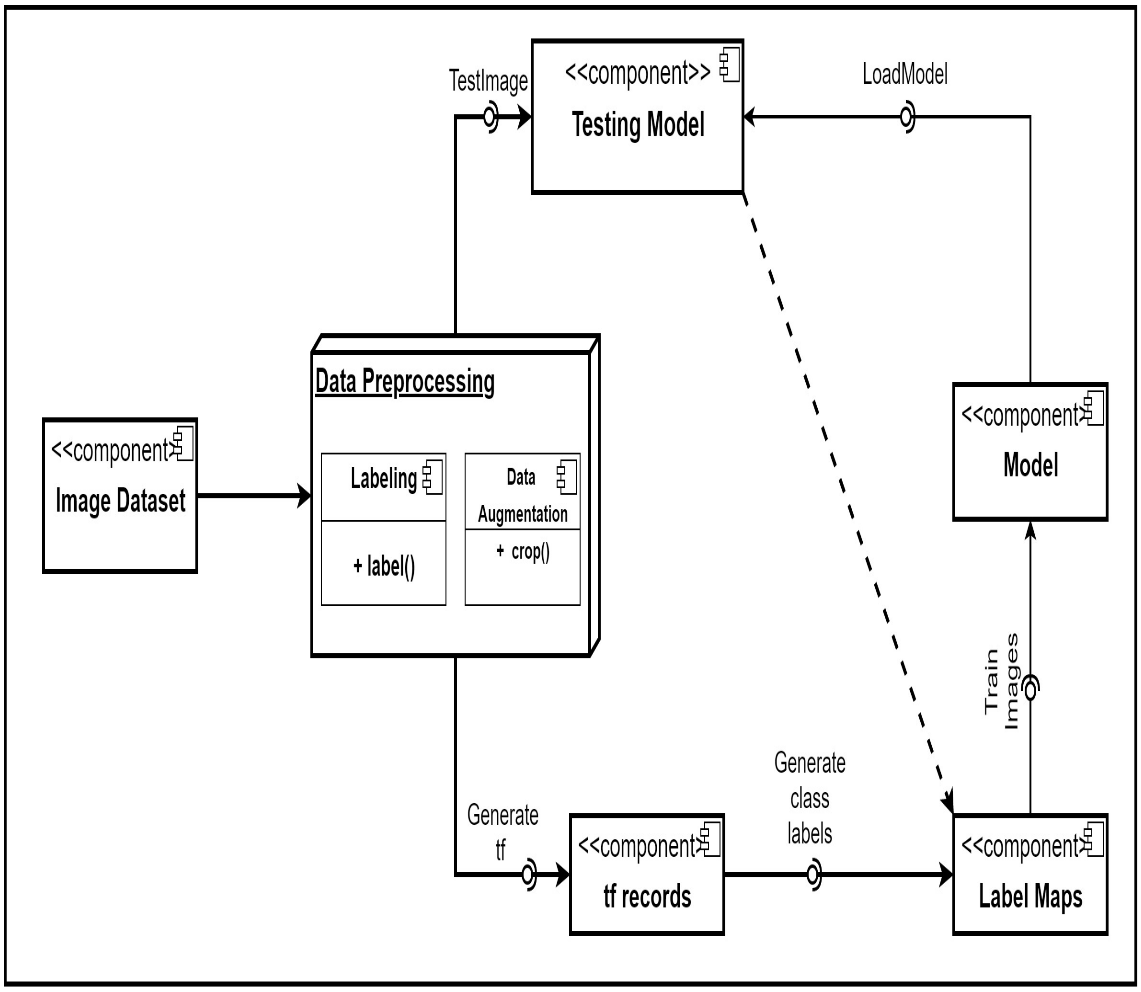


Figure 4.5: Component Diagram

4.3.5 Activity Diagram

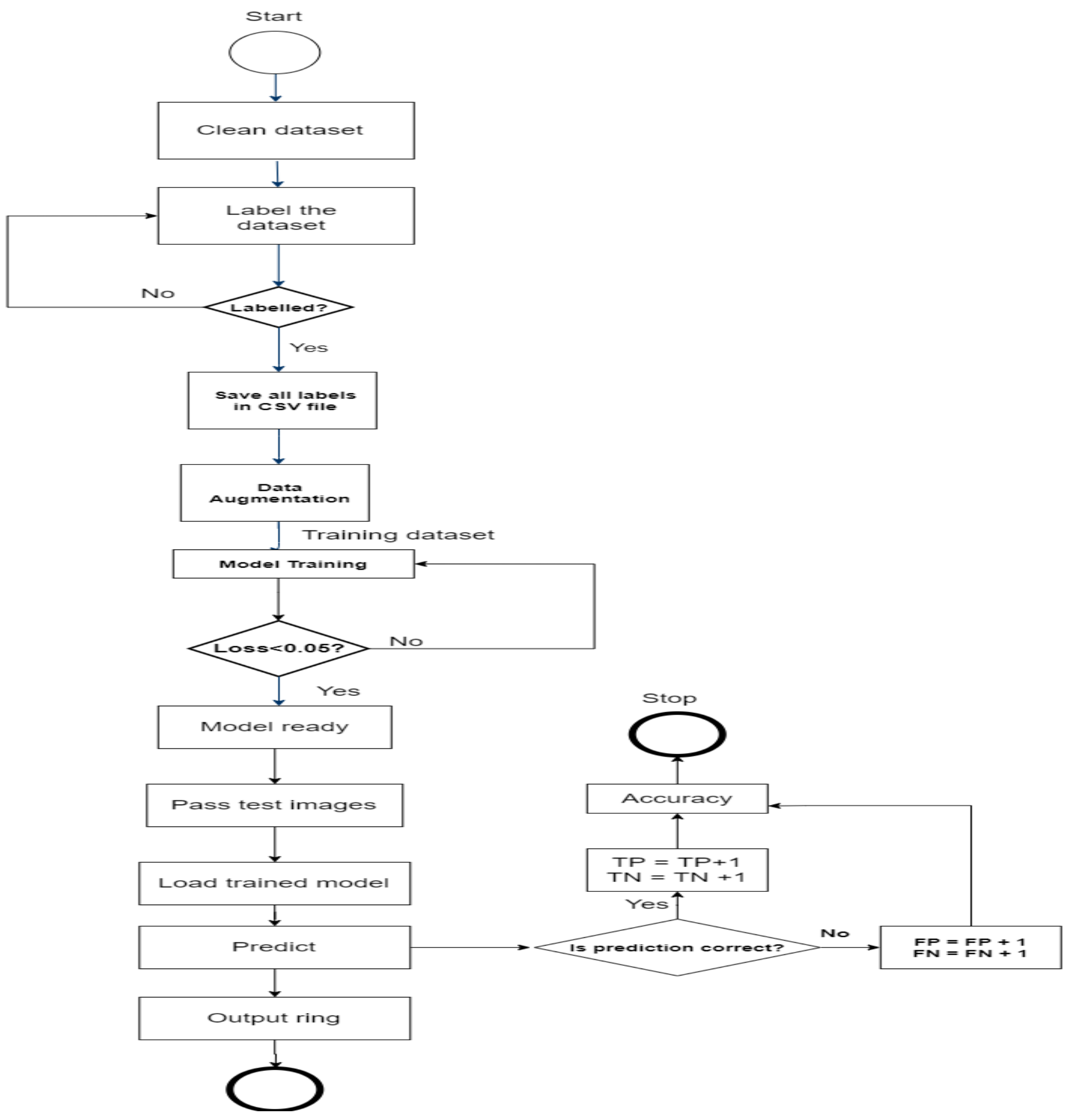


Figure 4.6: Activity Diagram

4.3.6 Communication Diagram

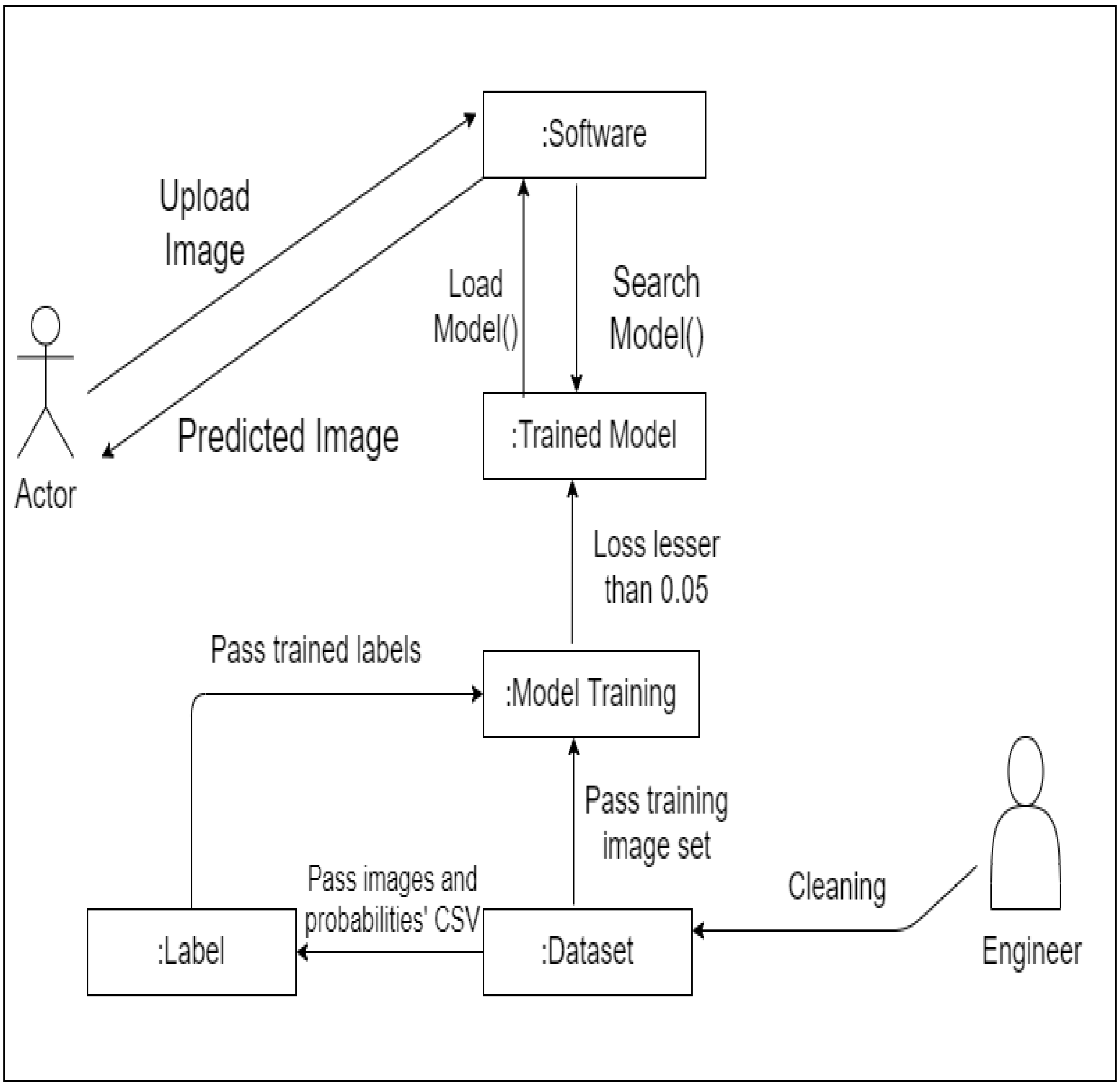


Figure 4.7: Communication Diagram

4.3.7 Sequence Diagram

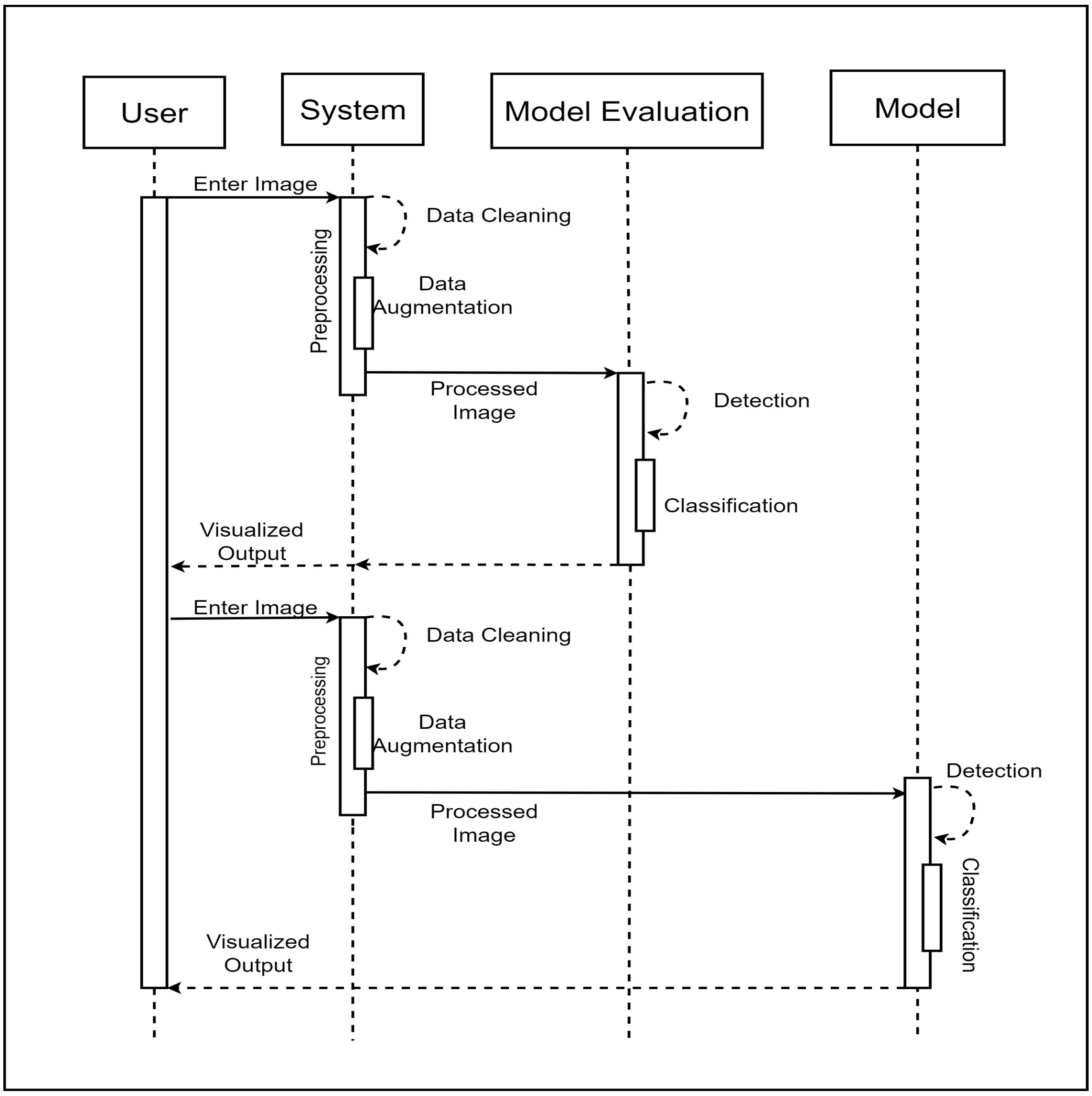


Figure 4.8: Sequence Diagram

4.4 Entity Relationship Diagram

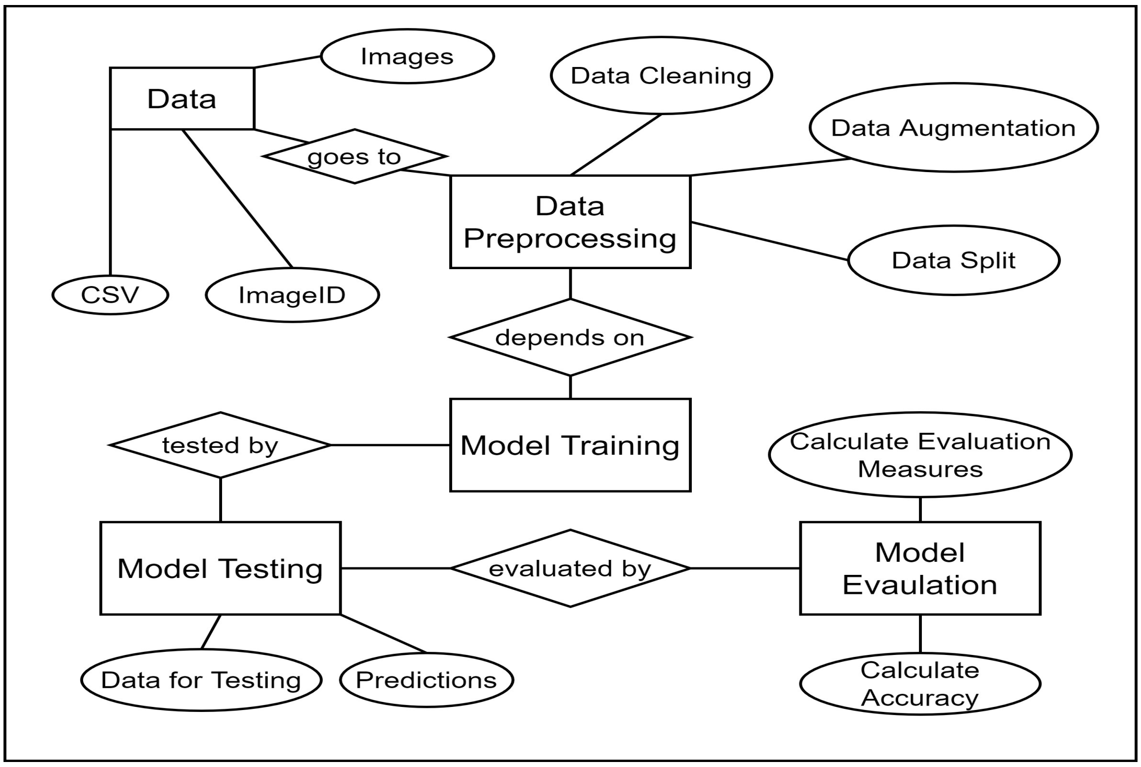


Figure 4.9: Entity Relationship Diagram

CHAPTER 5
PROJECT PLAN

5.1 Project Estimate

5.1.1 Reconciled Estimates

For every project, there has to be an estimated budget and it is always a topic of interest to the client to reconcile in order to cross check with contractor's estimation. As our project is purely software based, we are considering Lines of code for estimating the cost of system.

Total Lines of Code(LOC): 600
Time taken to complete: 3 months
LOC/month: 600/3 = 200

5.1.2 Project Resources

Journals and Reference Papers: The major references of our project work from IEEE and ACM.

5.2 Risk Management

5.2.1 Risk Identification

If unclean and unaugmented data is used, project suffers a huge risk of over-fitting.This was identified when the model got confused and classified galaxies wrongly.

5.2.2 Risk Analysis

1. From the process of Risk Analysis we concluded that classifying 7 types of galaxies is not feasible.

2. Classes Elliptical and Spiral had tens of thousands of images. Classes such as 'Merger' and 'Other' were confusing the model.

3. The reason for the same was 'Merger' class has some merging galaxies and 'Other' class contained images having spacial anomalies.

5.2.3 Overview of Risk Mitigation, Monitoring, Management

1. The solution to the problem encountered was to exclude these classes.

2. These classes only had a few hundred images whereas the classes such as ’Elliptical’ & ’Spiral’ had tens of thousands images.

3. The data for these classes was also not uniform. It contained a few broken images.

5.3 Project Schedule

5.3.1 Project Task Set

Project Tasks		
Sr. No.	Activity Scheduled	Date
1	Topic Finalization	10/07/2019
2	Literature Survey	20/07/2019
3	Project Review 1	29/07/2019
4	Comparative Study and Drawbacks	30/07/2019
5	Finalization of Algorithm	25/08/2019
6	Project Review 2	20/09/2019
7	Preparation of Preliminary Report	17/10/2019
8	Installation	27/12/2019
9	Pre-processing of dataset	10/01/2020
10	Data Augmentation	23/01/2020
11	Model Training	05/02/2020
12	Model Testing	20/03/2020
13	Project Completion with completed report	12/04/2020

Table 5.1: Project Tasks

5.3.2 Task Network

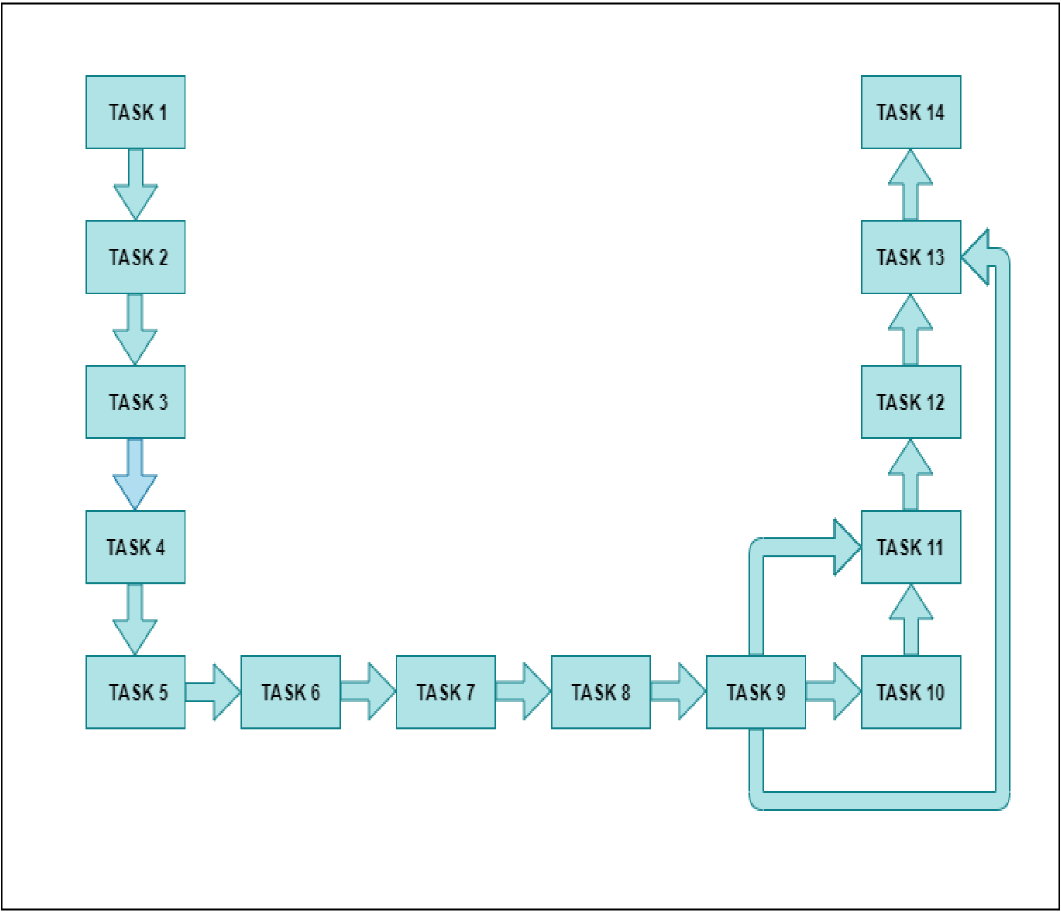


Figure 5.1: Task Network

5.3.3 Timeline Chart

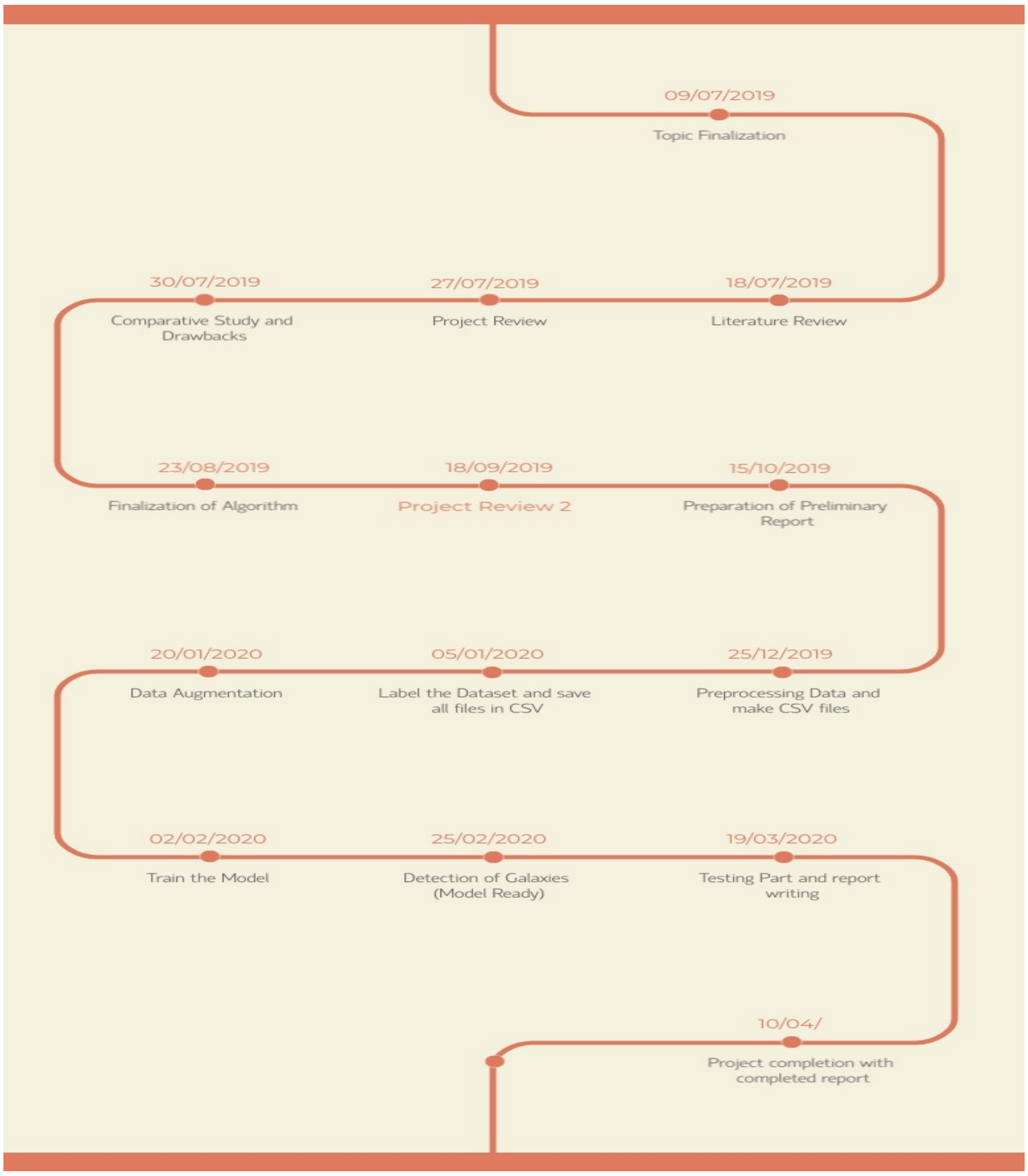

09/07/2019
Topic Finalization
30/07/2019
Comparative Study and Drawbacks
27/07/2019
Project Review
18/07/2019
Literature Review
23/08/2019
Finalization of Algorithm
18/09/2019
Project Review 2
15/10/2019
Preparation of Preliminary Report
20/01/2020
Data Augmentation
05/01/2020
Label the Dataset and save all files in CSV
25/12/2019
Preprocessing Data and make CSV files
02/02/2020
Train the Model
25/02/2020
Detection of Galaxies (Model Ready)
19/03/2020
Testing Part and report writing
10/04/
Project completion with completed report

Figure 5.2: Timeline Chart

5.4 Team Organization

5.4.1 Team Structure

The project team consist of four people, all of them studying in fourth year computer engineering and having expertise in machine learning and deep learning technologies.

- Tejshree Bang, has performed preprocessing of the data, data augmentation that is cropping, saved tf records before beginning the training and testing of model.
- Riya Chaddha, has performed installation and setting up environment for model, labelling of data, performed data augmentation on data and testing of model.
- Kimaya Devasthali, has prepared the dataset i.e. performed data cleaning, data augmentation solved the problem of overfitting during data training.
- Prasad Katore, has performed installation and setting up environment, generated tf records of data, label maps of the galaxy classes, trained and tested the model.

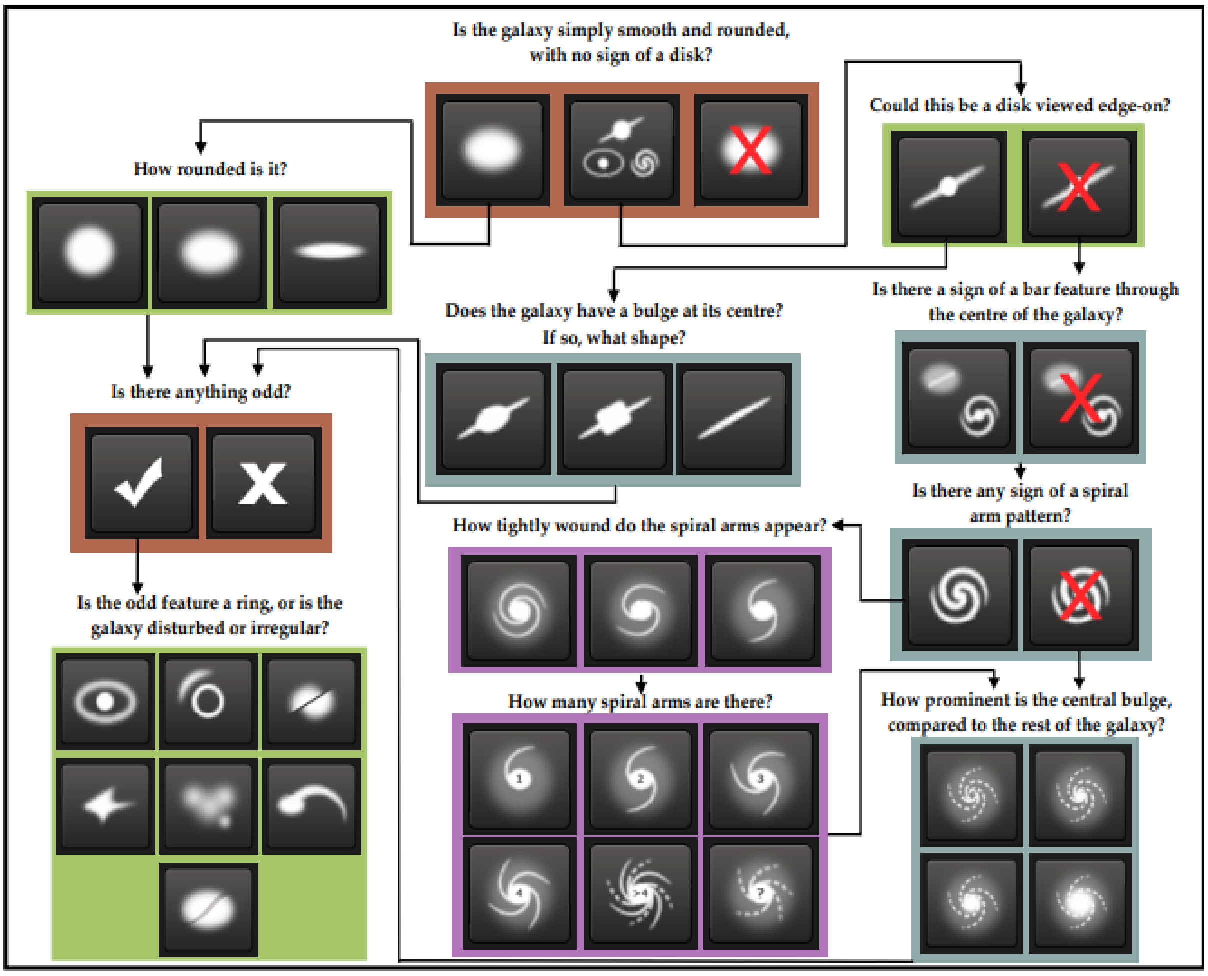
5.4.2 Management reporting and communication

1. Project Guide : Mrs. Archita Patil
2. Project Co-ordinator : Prof. Deipali Gore
3. Reporting : On every Monday, we used to report to our guide and discuss algorithm approaches, results and problems of project and also every step of the implementation.
4. Communication : Conduction of project reviews helped us communicate with the industry experts and their inputs and suggestions helped us in our project.

CHAPTER 6
PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

Image Pre-processing module:
All the images are originally unlabelled. These images are labelled using class probabilities which consists of probabilities for every class of galaxy where the class having maximum probability is chosen as a class label for that particular image. This is achieved by preparing a program having a decision tree which will assist the process of class label for many such unlabelled images.

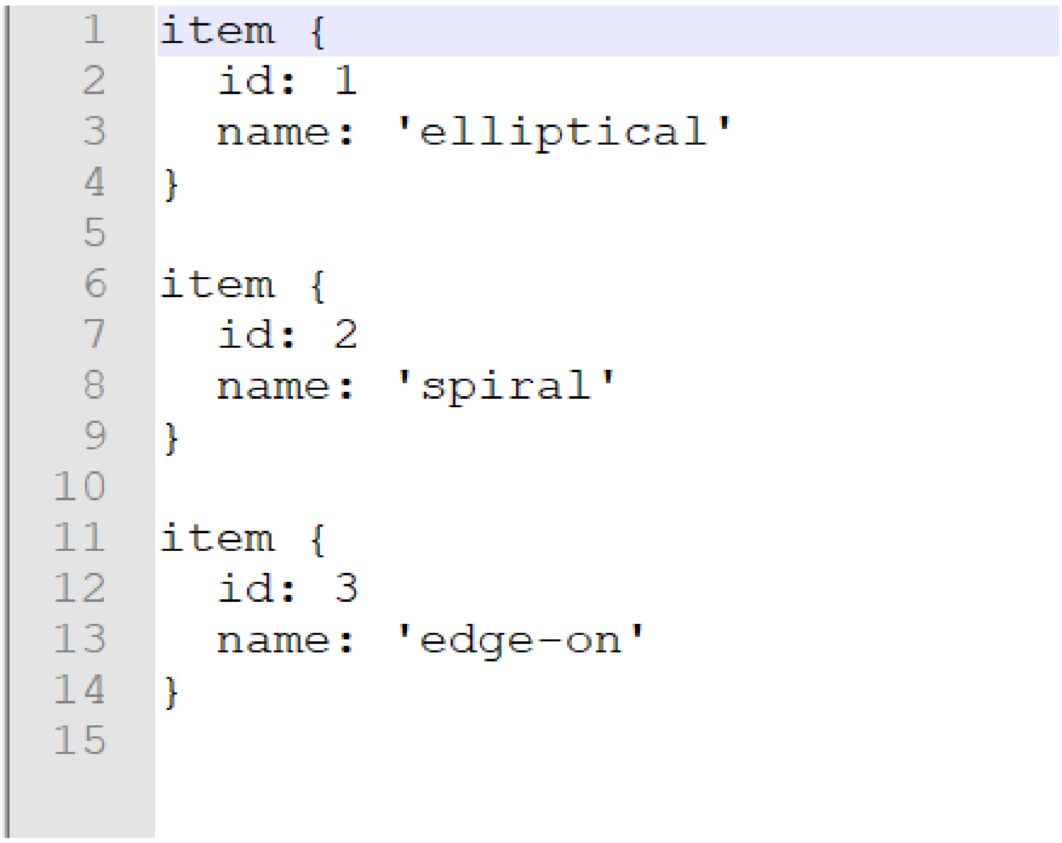


Is the galaxy simply smooth and rounded, with no sign of a disk?
Could this be a disk viewed edge-on?
How rounded is it?
Does the galaxy have a bulge at its centre? If so, what shape?
Is there a sign of a bar feature through the centre of the galaxy?
Is there anything odd?
Is there any sign of a spiral arm pattern?
How tightly wound do the spiral arms appear?
Is the odd feature a ring, or is the galaxy disturbed or irregular?
How many spiral arms are there?
How prominent is the central bulge, compared to the rest of the galaxy?

Figure 6.1: Decision Tree

Figure 6.1 demonstrates how the actual labels are assigned using class probabilities. For the 1st question, there are three answers, so for first answer it will go to the second question. Similarly, for second answer it will go to the third question and for third answer, the loop will terminate. To choose the correct answer, class probabilities are used where they indicate which answer has the maximum probability of being correct based on the responses recorded from the scientists.

Training module:
In this module, a CSV file is generated that consists of image id, class name and feature dimensions i.e. xmin, ymin, xmax, ymax. For this CSV file, tf records are generated. Tf records are basically input given to model in binary format. A pbtx file is created to store and save all the class labels as shown in the figure 6.2 below.



```
item {
  id: 1
  name: 'elliptical'
}

item {
  id: 2
  name: 'spiral'
}

item {
  id: 3
  name: 'edge-on'
}
```

Figure 6.2: Label Maps

This file stores label maps for the data that are class id and class name. Next the model is trained for several hours using Faster RCNN algorithm API.

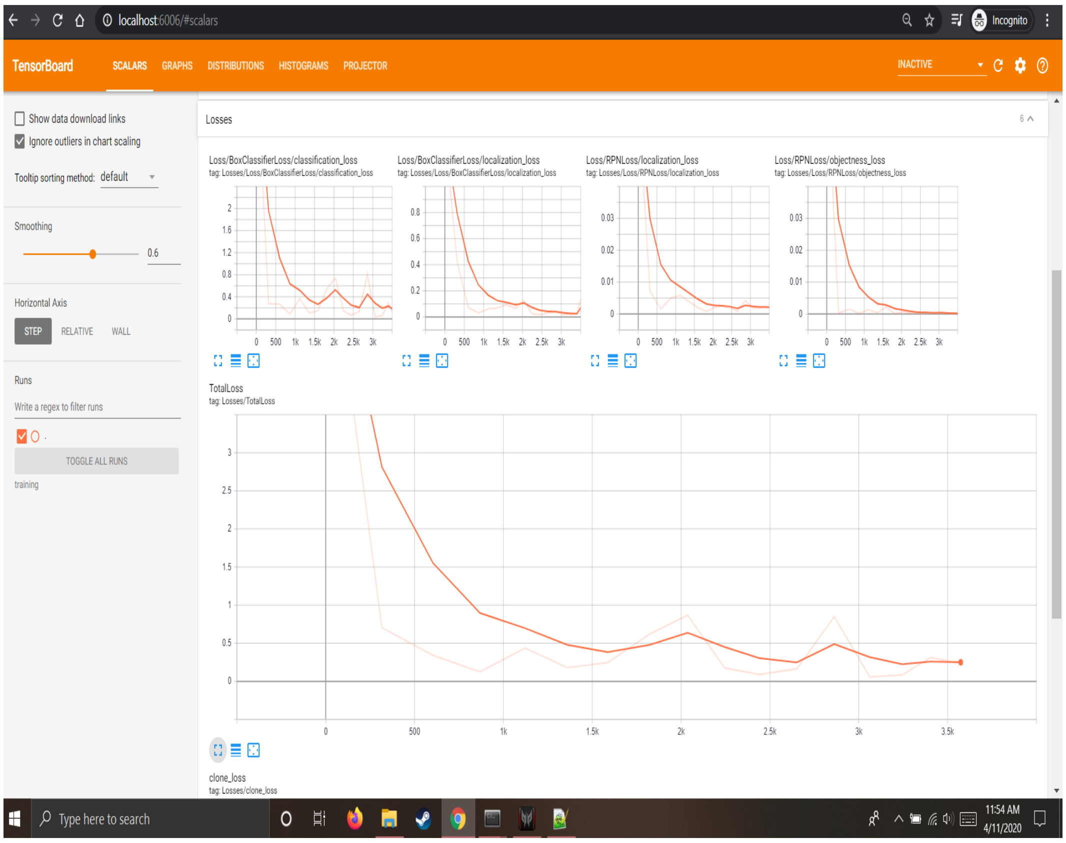
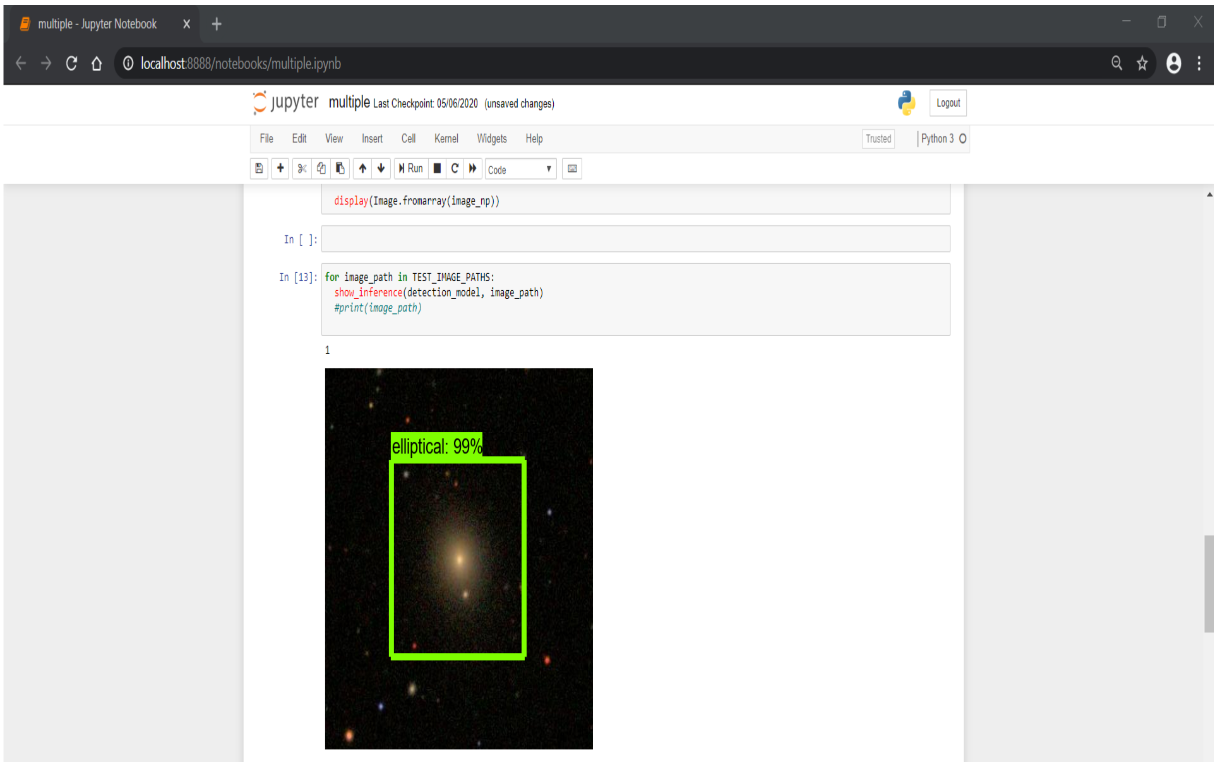

localhost:6006/#scalars
Incognito
TensorBoard
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
PROJECTOR
INACTIVE
Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
training
Losses
Loss/BoxClassifierLoss/classification_loss
tag: Losses/Loss/BoxClassifierLoss/classification_loss
Loss/BoxClassifierLoss/localization_loss
tag: Losses/Loss/BoxClassifierLoss/localization_loss
Loss/RPNLoss/localization_loss
tag: Losses/Loss/RPNLoss/localization_loss
Loss/RPNLoss/objectness_loss
tag: Losses/Loss/RPNLoss/objectness_loss
TotalLoss
tag: Losses/TotalLoss
0
0.5
1
1.5
2
2.5
3
500
1k
1.5k
2k
2.5k
3k
3.5k
clone_loss
tag: Losses/clone_loss
Type here to search
11:54 AM
4/11/2020

Figure 6.3: Total Loss Graph

Figure 6.3 shows the graph for total loss while training model. It can be seen that the loss is gradually decreasing after few hours of training and the process of training is stopped when the graph came to a point where loss was less than 0.05%. An inference graph gets generated for the model that is trained.

Testing module:
For this module, we load the trained model that is saved in the inference graph file. For testing the model, 1400 images are used for testing which are separated from the main dataset. For testing model, a code is generated by importing required libraries and while testing the images a path to the testing images is given in the code. For the images to be classified, trained model needs to be called and also the image label file is loaded in the test file.


multiple - Jupyter Notebook
localhost:8888/notebooks/multiple.ipynb
jupyter multiple Last Checkpoint: 05/06/2020 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
elliptical: 99%

```
        display(Image.fromarray(image_np))

In [ ]:

In [13]: for image_path in TEST_IMAGE_PATHS:
           show_inference(detection_model, image_path)
           #print(image_path)

         1
```

Figure 6.4: Test Output

Refer figure 6.4, the image shows detected and classified image which shows bounding box around the detected galaxy along with it's label and correctness of model in prediction of class. To evaluate the model, accuracy is found out using accuracy_score() function in built in tensorflow.

6.2 Tools and Technologies Used

1. Windows OS
2. CUDA Framework i.e. CUDA 9.0 cuDNN 7.4
3. Anaconda Platform, specifically as follows :
   (a) Anaconda Command Promt for Installation.
   (b) Jupyter Notebook for writing code and running scripts.
   (c) JupyterLab as a Development Environment.
4. Libraries: Python packages: Tensorflow, Sklearn, Pandas, Numpy, Pillow, Pandas, Matplotlib, ntpath, csv, glob.

6.3 Algorithm Details

6.3.1 Overview of Faster RCNN

1. RCNN & Fast RCNN uses selective search to find out the region proposals.
2. Selective search is a slow and time-consuming process.
3. So they came up with an object detection algorithm that eliminates the selective search algorithm and lets the network learn the region proposals.
4. The image is provided as an input to a convolutional network which provides a convolutional feature map.
5. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals.
6. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

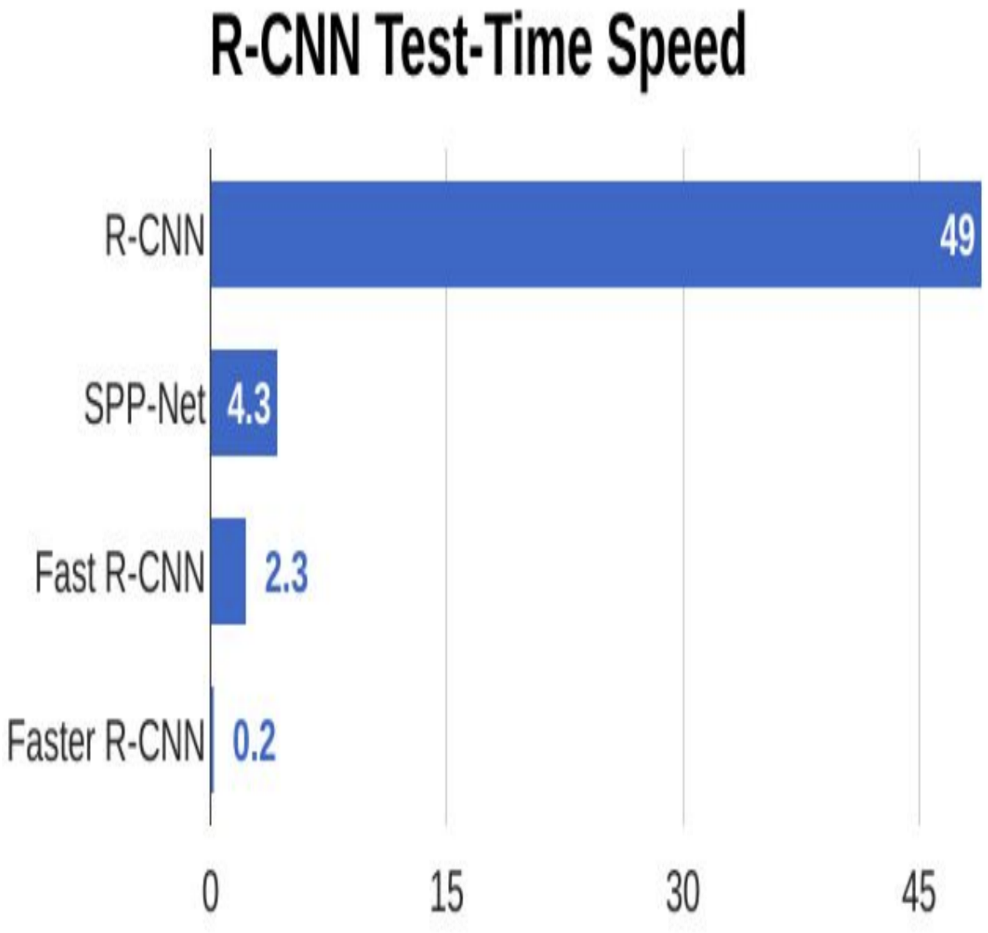


Figure 6.5: Comparison of RCNN

7. From the above graph, you can see that Faster R-CNN is much faster than it's predecessors. Therefore, it can even be used for real-time object detection.

6.3.2 Working of Faster RCNN

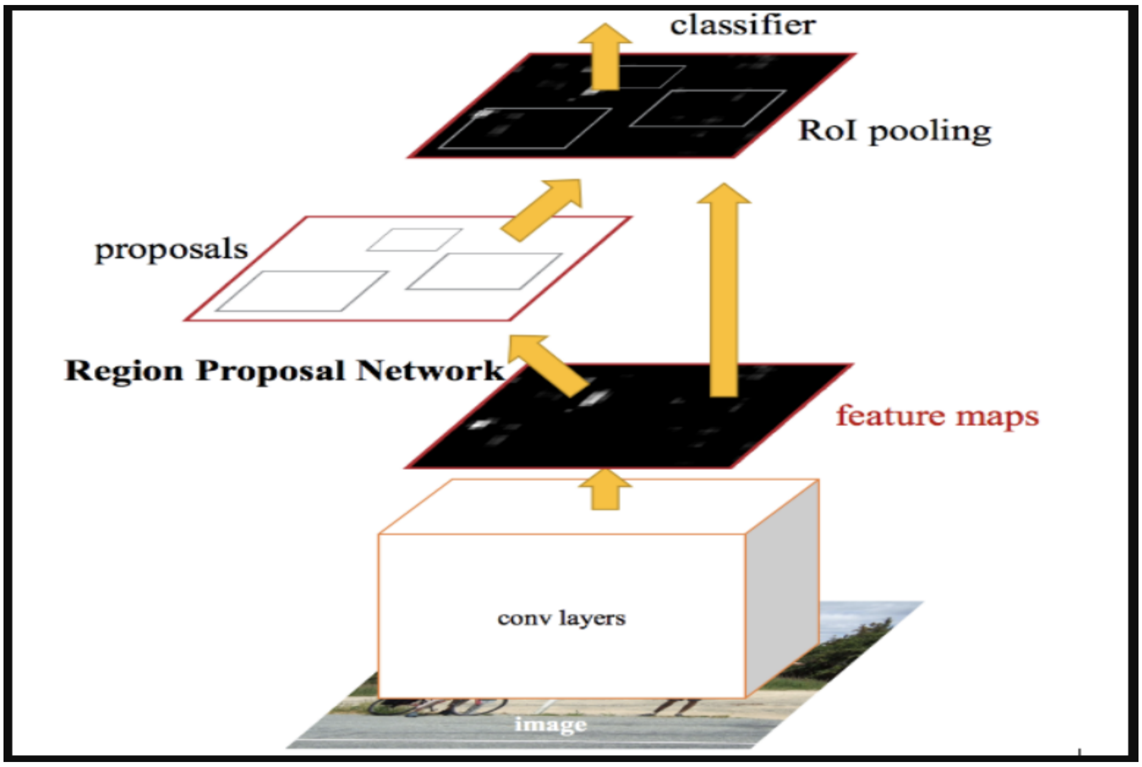


Figure 6.6: Block Diagram

1. Faster RCNN takes the feature maps from CNN and passes them on to the Region Proposal Network. RPN uses a sliding window over these feature maps, and at each window, it generates k Anchor boxes of different shapes and sizes.

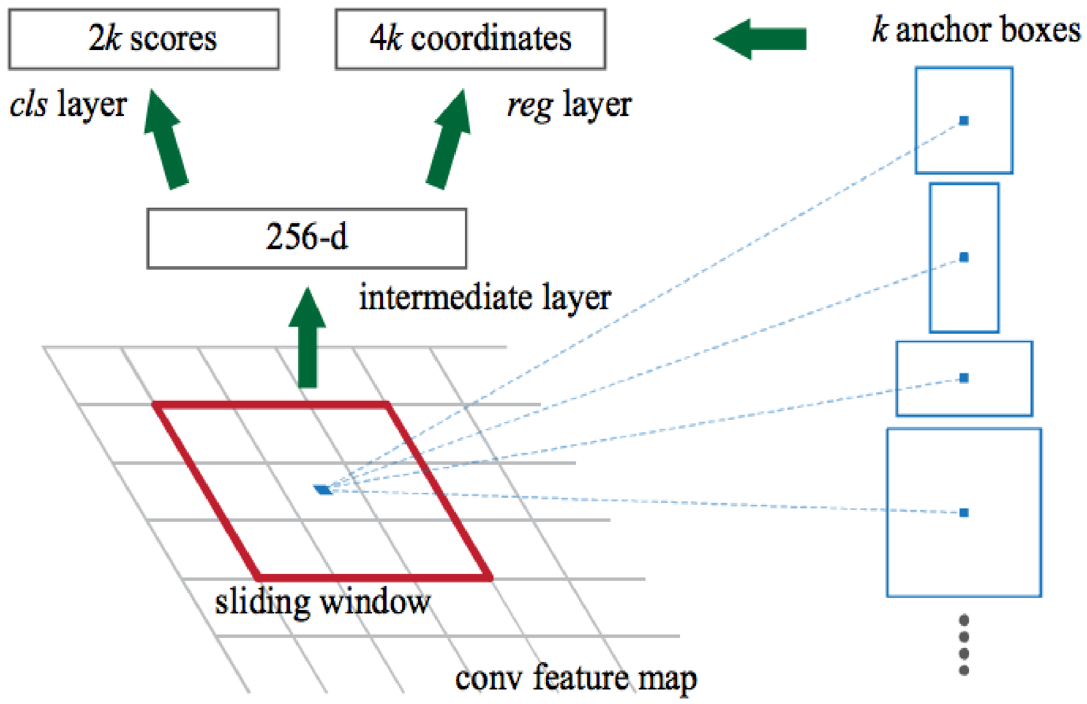


Figure 6.7: Generation of anchor boxes

2. Anchor boxes are fixed sized boundary boxes that are placed throughout the image and have different shapes and sizes. For each anchor, RPN predicts two things:
   a) The first is the probability that an anchor is an object.
   b) Second is the bounding box regressor for adjusting the anchors to better fit the object.

3. We now have bounding boxes of different shapes and sizes which are passed on to the RoI pooling layer. Now it might be possible that after the RPN step, there are proposals with no classes assigned to them.

4. We can take each proposal and crop it so that each proposal contains an object. This is what the RoI pooling layer does. It extracts fixed sized feature maps for each anchor.

5. Then these feature maps are passed to a fully connected layer which has a softmax and a linear regression layer.

6. It finally classifies the object and predicts the bounding boxes for the identified objects.

6.3.3 Region Proposal Network(RPN)

1. Region Proposal Network is the key peculiarity of this algorithm. The region proposal network (RPN) starts with the input image being fed into the backbone convolutional neural network. The input image is first resized such that it's shortest side is 600px with the longer side not exceeding 1000px.

2. The output features of the backbone network are usually much smaller than the input image depending on the stride of the backbone network. For both the possible backbone networks (VGG, ZF-Net) the network stride is 16. This means that two consecutive pixels in the backbone output features correspond to two points 16 pixels apart in the input image.

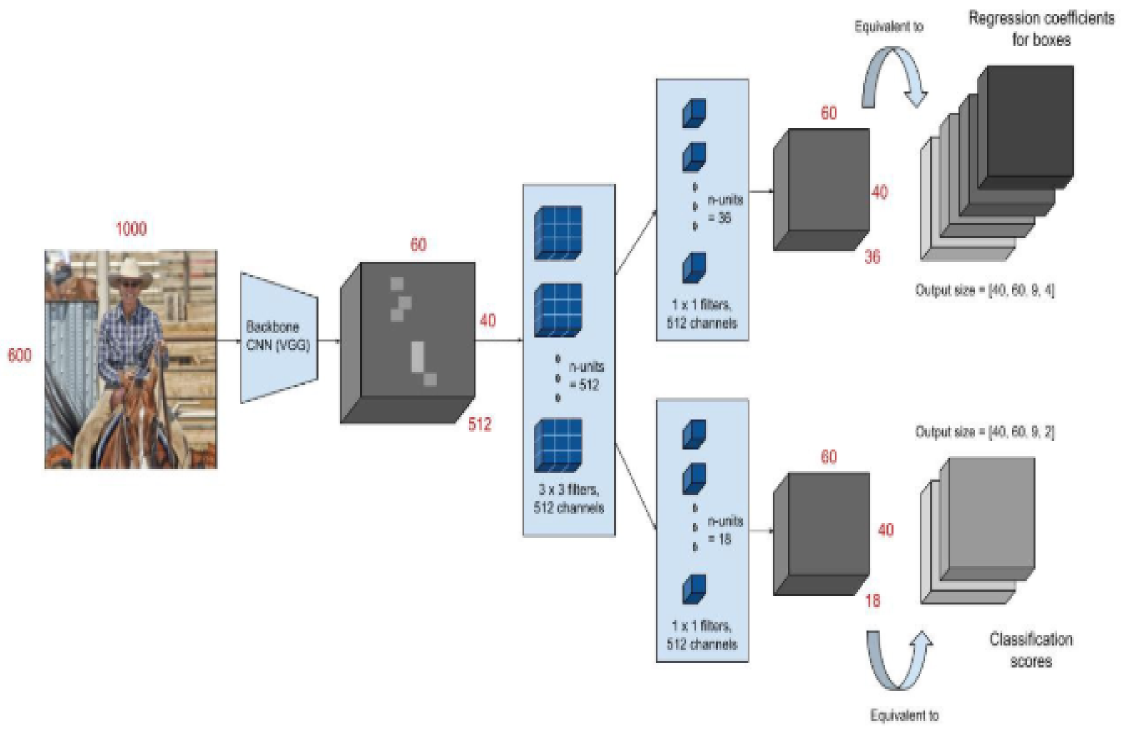

Equivalent to
Regression coefficients for boxes
60
40
36
n-units = 36
1 x 1 filters, 512 channels
Output size = [40, 60, 9, 4]
1000
600
Backbone CNN (VGG)
60
40
512
n-units = 512
3 x 3 filters, 512 channels
Output size = [40, 60, 9, 2]
60
40
18
n-units = 18
1 x 1 filters, 512 channels
Classification scores
Equivalent to

Figure 6.8: Architecture of RPN

3. For every point in the output feature map, the network has to learn whether an object is present in the input image at its corresponding location and estimate its size. This is done by placing a set of “Anchors” on the input image for each location on the output feature map from the backbone network. These anchors indicate possible objects in various sizes and aspect ratios at this location.

CHAPTER 7

SOFTWARE TESTING

7.1 Types of Testing

1. Unit Testing : Test the Model on a single image and a single class of images to check whether is gives accurate results without workload.
2. Module Testing : Check a single class of Images for the Model to assure that model has not overfitted.
   Ensure that Data Cleaning phase has occurred properly by ensuring image classification is accurate. This was done on multiple Computers with different data each time to attain max accuracy.
3. Integration Testing : Testing how the System works against different parameters of Data Augmentation and then training the testing the model. This was done on a small sample of data before applying to whole dataset.
4. System Testing : Testing the whole system from Data Preprocessing phases to Model Evaluation to obtain expected results.

7.2 Test Cases & Test Results

1. Unit Testing:
   Unit Testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Sr. No.	Test Case	Pre condition	Test data	Expected Output	Post condition	Actual Output	Result
1	Load model	Model should be created	Model	Model loaded	Model detected	Model loaded	Pass
2	Search images	Image should be saved in directory	Images	Images found	Images detected	Images found	Pass

Table 7.1: Unit Testing

2. Integration Testing
   Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Integration Testing is the second level of testing performed after Unit Testing and before System Testing.

Sr. No.	Test Case	Pre condition	Test data	Expected Output	Actual Output	Post condition	Result
1	Check if all installations are done properly	Install all the required libraries	Pre trained Model	Pre trained model runs as expected	Pre trained model detected	Pre trained model runs as expected	Pass

Table 7.2: Integration Testing

3. System Testing
   System testing is testing a complete and integrated software. The purpose of this test is to evaluate the system's compliance with the specified requirements. The process of testing an integrated system to verify that it meets specified requirements.

Sr. No.	Test Case	Pre condition	Test data	Expected Output	Post condition	Actual Output	Result
1	Check prediction done by system	Load trained model s	Input Image	Edge-on	Galaxy detected	Edge-on	Pass

Table 7.3: System Testing

CHAPTER 8

RESULTS

8.1 Outcomes

Evaluation of system is necessary to find out how efficient and good the system is built. There are various parameters on basis of which we can find out the accuracy of a system such as accuracy, precision, recall, root mean square error(RMSE), confusion matrix etc. We have calculated accuracy to evaluate classification model.

For evaluating a classification model, accuracy or confusion matrix is used. Accuracy is calculated on the basis of correctly classified images. Formula for accuracy is,

Accuracy = (TP+TN)/(TP+FP+TN+FN)

where TP, TN, FP, FN represent the number of True Positives, True Negatives, False Positives and False Negatives respectively.

The table below shows the test results for test dataset:

Total images tested	Predicted Correct	Accuracy	Accuracy Percent
1400	1186	0.844128	84.41%

Table 8.1: Test Results

The dataset was split into 90% for training and 10% for testing. As shown in the table above, 1400 images were tested out of which 1186 were classified correctly giving us the accuracy of model as 84.41%. Also, the model gives a speed of 200 millisecond i.e. 0.2 seconds to process the image and output the predicted results.

8.2 Screen Shots

Prediction:

1. Elliptical:

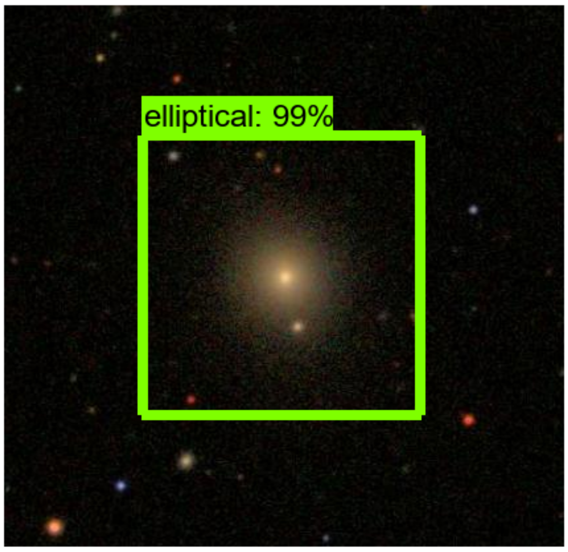


Figure 8.1: Elliptical Galaxy

2. Spiral:

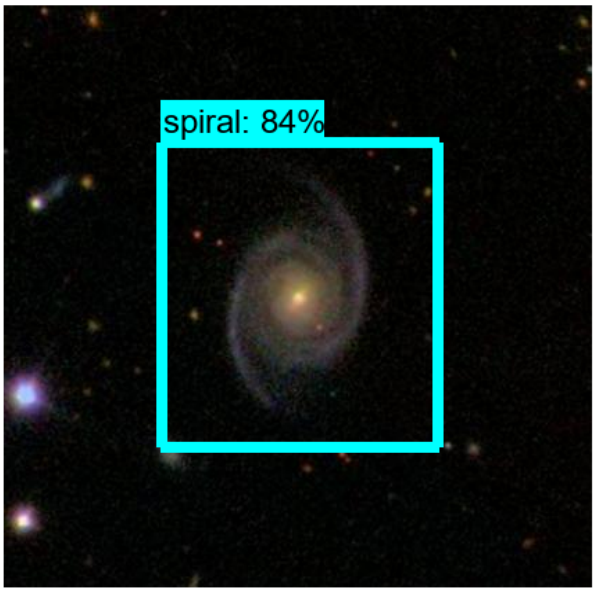


Figure 8.2: Spiral Galaxy

3. Edge-on :

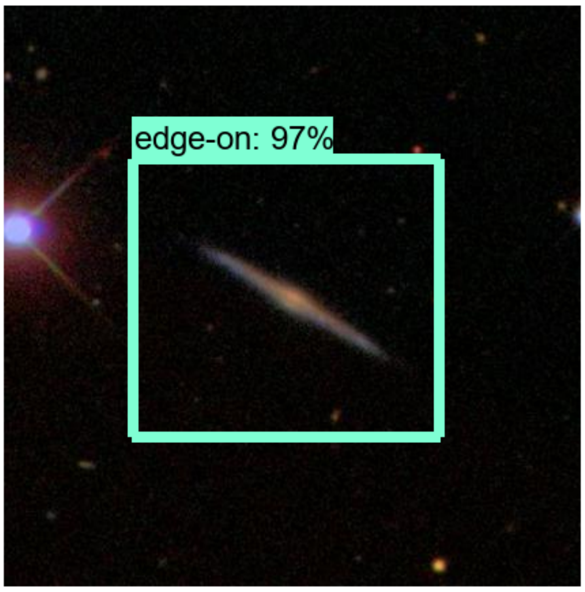
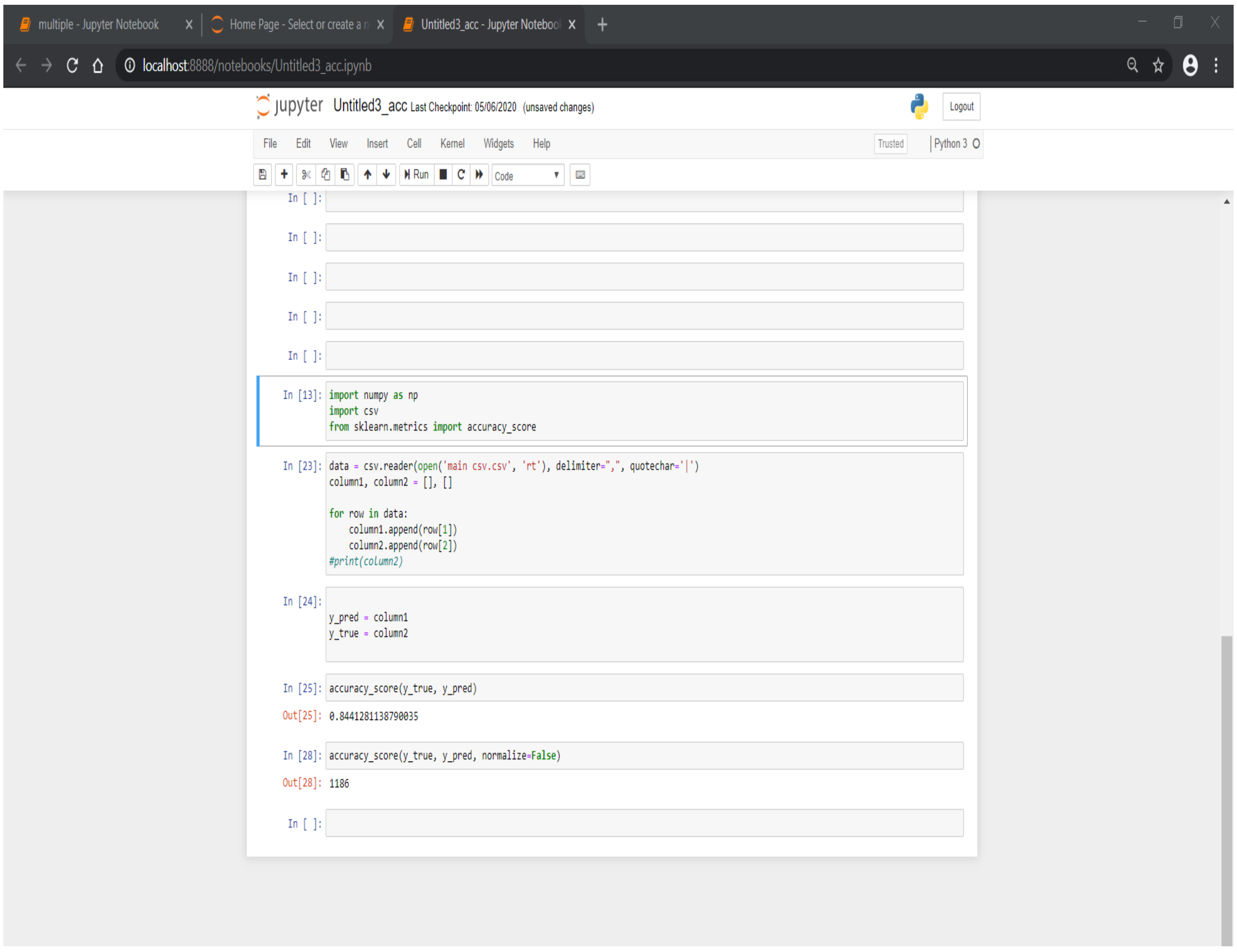


Figure 8.3: Edge-on Galaxy

Accuracy of model:


multiple - Jupyter Notebook
Home Page - Select or create a n
Untitled3_acc - Jupyter Notebook
localhost:8888/notebooks/Untitled3_acc.ipynb
jupyter Untitled3_acc Last Checkpoint: 05/06/2020 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [13]: import numpy as np
         import csv
         from sklearn.metrics import accuracy_score

In [23]: data = csv.reader(open('main csv.csv', 'rt'), delimiter=",", quotechar='|')
         column1, column2 = [], []

         for row in data:
             column1.append(row[1])
             column2.append(row[2])
         #print(column2)

In [24]: 
         y_pred = column1
         y_true = column2

In [25]: accuracy_score(y_true, y_pred)

Out[25]: 0.8441281138790035

In [28]: accuracy_score(y_true, y_pred, normalize=False)

Out[28]: 1186
```

Figure 8.4: Evaluation of model

CHAPTER 9

CONCLUSIONS

9.1 Conclusions

It is projected that SDSS will take 50 million images of galaxies in the near future. As a result it becomes utterly important to classify galaxies for relevant study. For this purpose we would be using Faster RCNN, a deep learning method which is more accurate than other object detectors. The dataset used for the project have been constructed from the public Data Releases of the Sloan Digital Sky Survey (SDSS). Our classification is based on the Hubble Morphological Classification of Galaxies where three main types of galaxies are Elliptical, Spiral and Edge-on. Time Performance would be affected largely when using a lower end GPU. Our project will be able to detect the type of galaxy in about 50 milliseconds per image on a NVIDIA GeForce GTX 1050 Ti GPU.

9.2 Future Work

1. Classifying other types of galaxies viz. Merger, Barred Spiral, Ring, Irregular.
2. In future, we could detect galaxies that are overlapped and segregate each individual galaxy from it.
3. Stratification on the data in order to avoid overfitting while training for those types of galaxies that are less in number.

9.3 Applications

1. Astronomers or astrophysicists study the shapes and nature of galaxies. This system will help them to conduct their research on galaxies so that they can predict the location and nature of galaxy.
2. Sky survey departments can automate the process of filtering and classifying large amount of images taken daily for the purpose of sky survey.

ANNEXURE A

Problem Analysis :

1. Feasibility Study :
   (a) Technical Feasibility : Project will be implemented using Python as a programming language. Dataset will be converted csv using ImageCn software. Deep learning methods are used to train the model. Using a high end GPU results are produced fast.
   (b) Schedule Feasibility : The project will be executed in stages. First stage is converting the dataset to csv file which will be done using aforementioned software ImageCN. Second stage is splitting the dataset for training and testing. Stage three is training the model with training dataset. Stage four is defining the type of galaxy for each image in testing dataset.
   (c) Economic Feasibility : The project requires a high end GPU for faster implementation so that it generates results in milliseconds.
   (d) Ethical Feasibility : The dataset used for the project is constructed from the Sloan Digital Sky Survey. Data releases by the SDSS are completely public. The algorithms used in the project are standardized and ethical.

2. Type of Problem :

   Problem Statement : To Detect and Classify Galaxies using Faster Region-based Convolutional Neural Networks (FasterRCNN).

   The problem statement implies that the Faster RCNN will work upon dataset of galaxy images to detect the galaxy in the image. For every input submitted to the algorithm it will be able to classify it according to the morphological classification system. Since there are specific types of galaxies the system will always be able to classify the galaxy.

   NP-Completeness applied to decision problems like this one pertains to the difficulty of the decision problem. The system will detect whether there is a Galaxy in the image and if there is one it will classify the galaxy according to the type.

   Thus the problem is NP-Complete.

ANNEXURE B

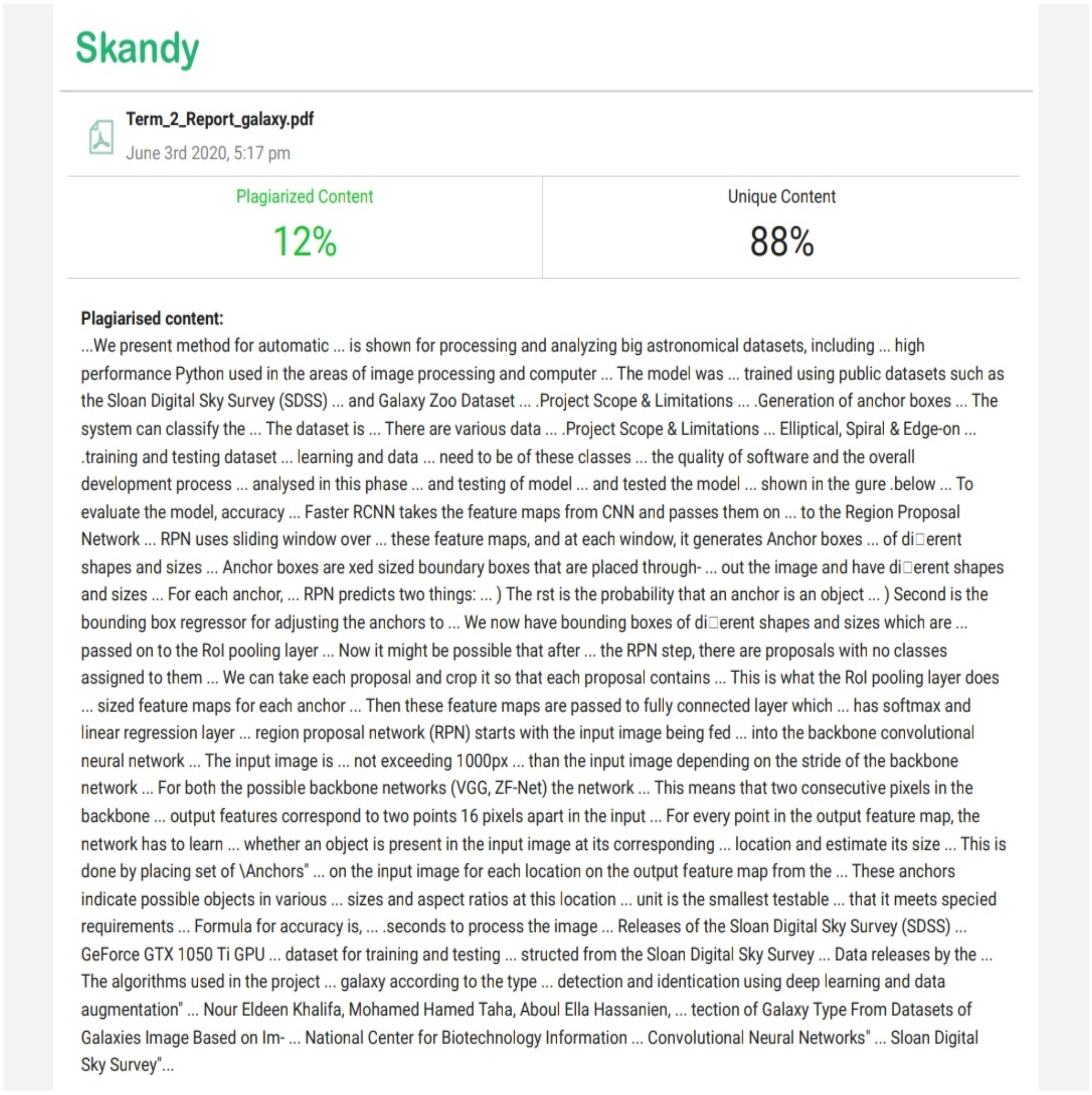
Paper Publication :

1. About Paper :
   Our paper is a literature review paper that provides information on the latest efforts in technologies used in classification of galaxies. We survey a few methods that use Neural Networks and Deep Convolutional Neural Networks alongside different approaches in data augmentation.

2. Publication :
   We submitted the paper to a few conferences but unfortunately it got rejected. Currently we are working in vastly improving the contents of the paper and making sure it provides significant insight on Machine Learning methods used in Galaxy detection and Classification.

ANNEXURE C

Plagiarism Report :



Skandy

Term_2_Report_galaxy.pdf

June 3rd 2020, 5:17 pm

Plagiarized Content

12%

Unique Content

88%

Plagiarised content:

...We present method for automatic ... is shown for processing and analyzing big astronomical datasets, including ... high performance Python used in the areas of image processing and computer ... The model was ... trained using public datasets such as the Sloan Digital Sky Survey (SDSS) ... and Galaxy Zoo Dataset ... .Project Scope & Limitations ... .Generation of anchor boxes ... The system can classify the ... The dataset is ... There are various data ... .Project Scope & Limitations ... Elliptical, Spiral & Edge-on ... .training and testing dataset ... learning and data ... need to be of these classes ... the quality of software and the overall development process ... analysed in this phase ... and testing of model ... and tested the model ... shown in the gure .below ... To evaluate the model, accuracy ... Faster RCNN takes the feature maps from CNN and passes them on ... to the Region Proposal Network ... RPN uses sliding window over ... these feature maps, and at each window, it generates Anchor boxes ... of di□erent shapes and sizes ... Anchor boxes are xed sized boundary boxes that are placed through- ... out the image and have di□erent shapes and sizes ... For each anchor, ... RPN predicts two things: ...) The rst is the probability that an anchor is an object ...) Second is the bounding box regressor for adjusting the anchors to ... We now have bounding boxes of di□erent shapes and sizes which are ... passed on to the RoI pooling layer ... Now it might be possible that after ... the RPN step, there are proposals with no classes assigned to them ... We can take each proposal and crop it so that each proposal contains ... This is what the RoI pooling layer does ... sized feature maps for each anchor ... Then these feature maps are passed to fully connected layer which ... has softmax and linear regression layer ... region proposal network (RPN) starts with the input image being fed ... into the backbone convolutional neural network ... The input image is ... not exceeding 1000px ... than the input image depending on the stride of the backbone network ... For both the possible backbone networks (VGG, ZF-Net) the network ... This means that two consecutive pixels in the backbone ... output features correspond to two points 16 pixels apart in the input ... For every point in the output feature map, the network has to learn ... whether an object is present in the input image at its corresponding ... location and estimate its size ... This is done by placing set of \Anchors" ... on the input image for each location on the output feature map from the ... These anchors indicate possible objects in various ... sizes and aspect ratios at this location ... unit is the smallest testable ... that it meets specied requirements ... Formula for accuracy is, ...seconds to process the image ... Releases of the Sloan Digital Sky Survey (SDSS) ... GeForce GTX 1050 Ti GPU ... dataset for training and testing ... structed from the Sloan Digital Sky Survey ... Data releases by the ... The algorithms used in the project ... galaxy according to the type ... detection and identication using deep learning and data augmentation" ... Nour Eldeen Khalifa, Mohamed Hamed Taha, Aboul Ella Hassanien, ... tection of Galaxy Type From Datasets of Galaxies Image Based on Im- ... National Center for Biotechnology Information ... Convolutional Neural Networks" ... Sloan Digital Sky Survey"...

Figure C.1: Plagiarism Report

CHAPTER 10
REFERENCES

[1] Roberto E. Gonzalez, Roberto P. Munoz, Cristian A. Hernandez, *“Galaxy detection and identification using deep learning and data augmentation”* arXiv.org, September 2018.

[2] Nour Eldeen Khalifa, Mohamed Hamed Taha, Aboul Ella Hassanien, Ibrahim Selim, *“Deep Galaxy V2: Robust Deep Convolutional Neural Networks for Galaxy Morphology Classifications”*. IEEE June 2018.

[3] Mohamed Abd El Aziz, I. M. Selim and Shengwu Xiong, *“Automatic Detection of Galaxy Type From Datasets of Galaxies Image Based on Image Retrieval Approach”*. National Center for Biotechnology Information U.S.A., Scientific Reports, June 30, 2017.

[4] Ricardo Contreras, Gabriel Salazar, M. Angelica Pinninghoff, Neil Nagar, *“Neural Networks on galaxy pattern recognition”*. 8th International Conference of Pattern Recognition Systems, 2017.

[5] Nour Eldeen Khalifa, Mohamed Hamed Taha, Aboul Ella Hassanien, Ibrahim Selim, *“Deep Galaxy: Classification of Galaxies based on Deep Convolutional Neural Networks”*. arXiv.org, September 2, 2017.

[6] Kyle Willet, Chris Lintott, Steven P. Bamford, Robert C. Nichol, Kevin Schawinski, Robert J. Simpson, Edward M. Edmondson, *”Galaxy Zoo 2: detailed morphological classifications for 304,122 galaxies from the Sloan Digital Sky Survey”*. arXiv:1308.3496v2 [astro-ph.CO], 19th August, 2013.

[7] Francois Chollet, *“Deep Learning with Python”*. - A book on deep learning by Google AI Researcher Francois Chollet.

[8] A Step By Step Introduction To The Basic Object Detection Algorithms Part 1, *“https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1/”*. - A post on implementation of object detection algorithms.

[9] Sloan Digital Sky Survey Data Release 15, *“https://www.sdss.org/dr15/”*. December 20, 2018.